Conditional Probability

This lab is an introduction to conditional probabilities.

The lab includes a visualization called an *icon array*. It's meant to be an instructional part of the lab to help build intuitions about conditional probability. These visualizations do not appear in the textbook and will not appear on any exam.

In [18]:
```
# Run this cell to set up the notebook, but please don't change it.

# These lines import the Numpy and Datascience modules.
import numpy as np
from datascience import *

# These lines do some fancy plotting magic.
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
import warnings
warnings.simplefilter('ignore', FutureWarning)

# This line loads the visualization code for this lab.
import visualizations

# These lines load the tests.
from client.api.notebook import Notebook
ok = Notebook('lab10.ok')
_ = ok.submit()
```

```
=====================================================================
Assignment: Conditional Probability
OK, version v1.14.20
=====================================================================



Saving notebook... No valid file sources found
Submit... 0.0% complete
Could not submit: Assignment does not exist
Backup... 0.0% complete
Could not backup: Assignment does not exist
```

1. What is conditional probability good for? ¶

Suppose we have a known population, like all dogs in California. So far, we've seen 3 ways of *predicting* something about an individual in that population, given incomplete knowledge about the identity of the individual:

- If we know nothing about the individual dog, we could predict that its speed is the *average* or *median* of all the speeds in the population.
- If we know the dog's height but not its speed, we could use *linear regression* to predict its speed from its height. The resulting prediction is still imperfect, but it might be more accurate than the population average.
- If we know the dog's breed, height, and age, we could use *nearest-neighbor classification* (or *multiple regression*) to predict its speed by comparing it to a collection of dogs with known speed.

We can also compute conditional probabilities to make predictions about individuals or events. This technique is different from the previous methods we've examined because

1. our prediction for each outcome is described by a probability, and
2. each probability can be exactly calculated from assumptions, as opposed to estimated from data.

2. Icon arrays

Parts 3 and 4 of this lab work with a more complex example about disease, but first, let's start with a simple example.

Imagine you are a marble. You don't know what you look like (since you obviously have no eyes), but you know that Samantha drew you **uniformly at random** from a bag that contained the following marbles:

- 4 large shiny marbles,
- 1 large dull marble,
- 6 small shiny marbles, and
- 2 small dull marbles.

Question 2.0.1. Knowing only what we've told you so far, what's the probability that you're a large shiny marble?

```
BEGIN QUESTION
name: q2_0_1
```

In [19]:
```
probability_large_shiny = 4/13 #SOLUTION
```

In [20]:
```
# TEST
np.isclose(probability_large_shiny, 4/13)
```

Out[20]:
```
True
```

In [21]:
```
# TEST
0 < probability_large_shiny < 1
```
Out[21]:
```
True
```

Here's a table with those marbles:

In [22]:
```
marbles = Table.read_table("marbles.csv")
marbles.show()
```

surface	size
shiny	large
shiny	large
shiny	large
shiny	large
dull	large
shiny	small
shiny	small
shiny	small
shiny	small
shiny	small
shiny	small
dull	small
dull	small

Here are the counts of each type of marble in a pivot table.

In [23]:
```
marbles.pivot('surface', 'size')
```
Out[23]:

size	dull	shiny
large	1	4
small	2	6

Here are all the different combinations of surface and size, with the count for each surface-size combination. Each type of marble appears in its own row.

```
In [24]: marbles.group(['surface', 'size'])
```

Out[24]:

surface	size	count
dull	large	1
dull	small	2
shiny	large	4
shiny	small	6

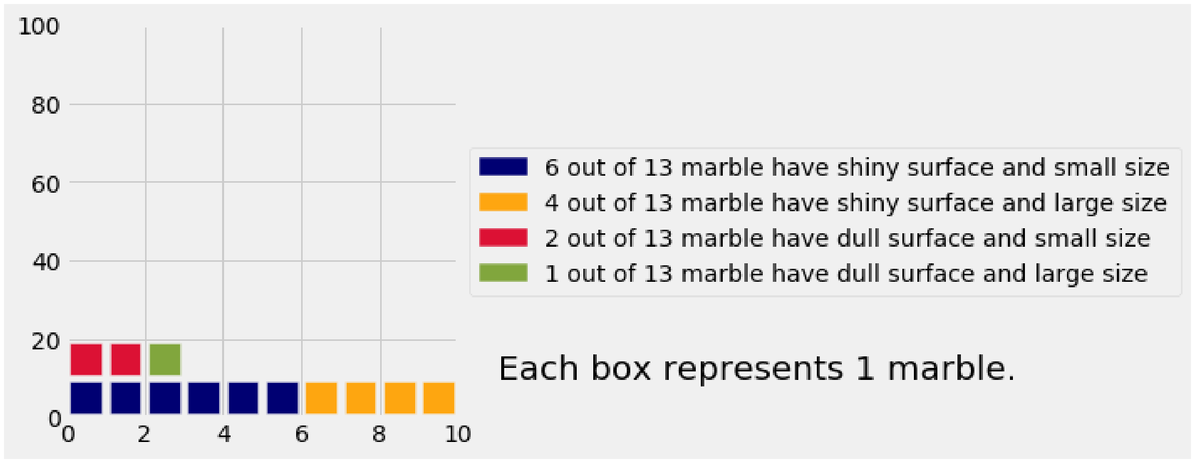
We've included some code to display something called an *icon array*. The functions in the cell below create icon arrays from various kinds of tables. Don't worry about understanding the code; just run this cell.

In [25]:
```
# Run this cell.

#######################################################################
#
# The functions you'll need to actually use are in here.  Each is a
# way of making an icon array from a differently-formatted table.
#######################################################################
#

def display_icon_array(table, groups, individuals_name):
    """
    Given a table and some columns to group it on, displays an icon ar
ray
    of the groups.

    groups should be an array of labels of columns in table.

    individuals_name is your name for the individual rows of table.
    For example, if we're talking about a population of people,
    individuals_name should be "people".

    For example:

    display_icon_array(marbles, ["surface", "size"], "marbles")
    """
    display_grouped_icon_array(table.group(groups), individuals_name)

def display_grouped_icon_array(grouped_data, individuals_name):
    """
    Given a table with counts for data grouped by 1 or more categorie
s,
    displays an icon array of the groups represented in the table.

    grouped_data should be a table of frequencies or counts, such as
    a table created by calling the groups method on some table.

    individuals_name is your name for the individual members of the
    dataset.  For example, if we're talking about a population of
    people, individuals_name should be "people".

    For example:

    display_grouped_icon_array(marbles.group(["surface", "size"]), "ma
rbles")
    """
    visualizations.display_combinations(grouped_data, individuals_name
=individuals_name)

def display_crosstab_icon_array(crosstabulation, x_label, individuals_
name):
    """
    Given a crosstabulation table, displays an icon array of the group
s
    represented in the table.

    crosstabulation should be a table of frequencies or counts created
```

```
by
    calling pivot on some table.

    x_label should be the label of the categories listed as columns (o
n
    the "x axis" when the crosstabulation table is printed).

    individuals_name is your name for the individual members of the
    dataset.  For example, if we're talking about a population of
    people, individuals_name should be "people".

    For example:

    display_crosstab_icon_array(marbles.pivot("surface", "size"), "sur
face", "marbles")
    """
    display_grouped_icon_array(visualizations.pivot_table_to_groups(cr
osstabulation, x_label), individuals_name)
```

Here's an icon array of all the marbles, grouped by surface and size:

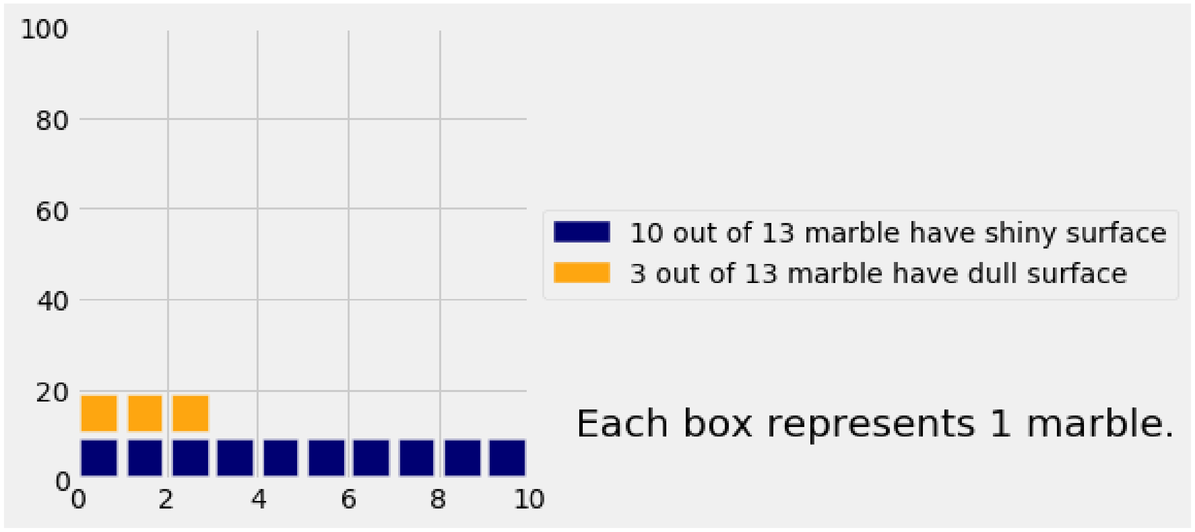
In [26]:
```
# Run this cell.
display_grouped_icon_array(marbles.group(["surface", "size"]), "marbl
e")
```



You (the marble) should imagine that you are a random draw from these 13 icons.

The following is an icon array of the marbles, grouped **only by their surface (shiny/dull)**.

```
In [27]: display_grouped_icon_array(marbles.group("surface"), "marble")
```



Knowing nothing else about yourself, you're equally likely to be any of the marbles pictured.

Question 2.0.2. What's the probability that you're a shiny marble? Calculate this by hand (using Python for arithmetic) by looking at your icon array.

```
BEGIN QUESTION
name: q2_0_2
```

```
In [28]: probability_shiny = 10/13 #SOLUTION
```

```
In [29]: # TEST
         np.isclose(probability_shiny, 10/13)
```

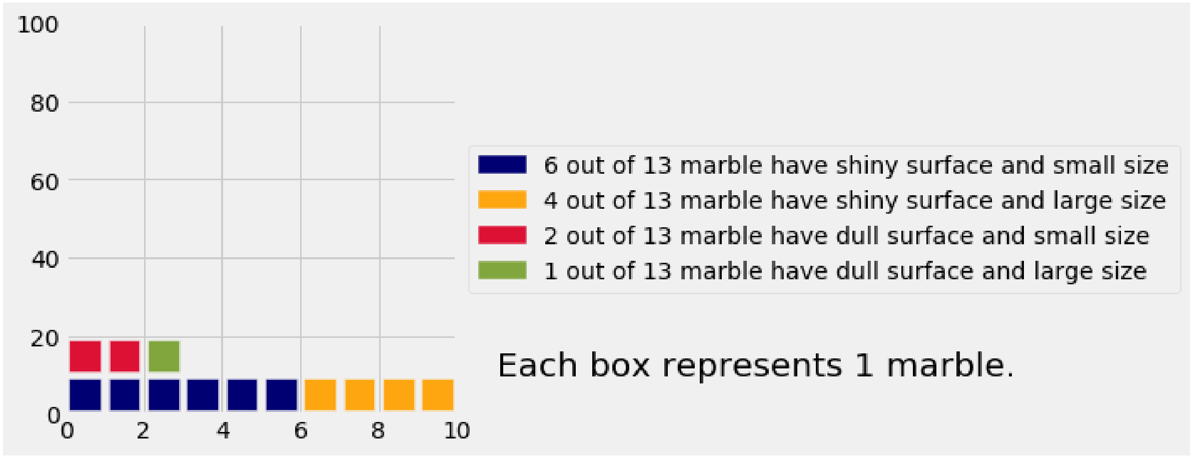
```
Out[29]: True
```

2.1. Conditional probability

Suppose you overhear Samantha say that you're a large marble. Does this somehow change the chance that you're shiny? Let's find out.

Go back to the full icon array, displayed below for convenience.

In [30]:
```
display_grouped_icon_array(marbles.group(["surface", "size"]), "marbl
e")
```

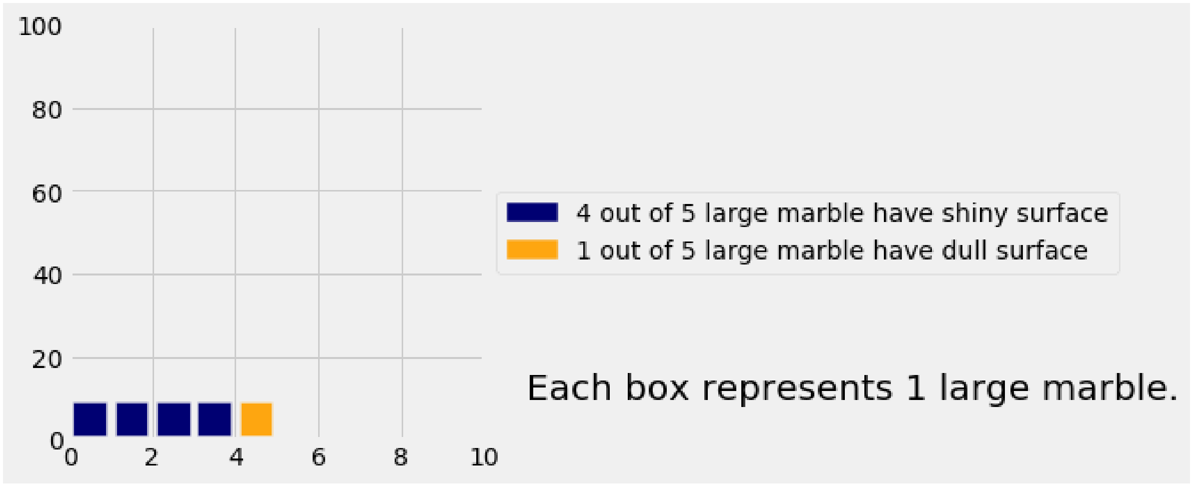


In question 2.0.2, we assumed you were equally likely to be any of the marbles, because we didn't know any better. That's why we looked at all the marbles to compute the probability that you were shiny.

But assuming that you're a large marble, we can eliminate some of these possibilities. In particular, you can't be a small shiny marble or a small dull marble.

You're still equally likely to be any of the remaining marbles, because you don't know any other information. So here's an icon array of those remaining possibilities:

In [31]:
```
# Just run this cell.
display_grouped_icon_array(marbles.where("size", "large").group("surfa
ce"), "large marble")
```



Question 2.1.1. What's the probability you're a shiny marble, knowing that you're a large marble?

Hint: Use the icon array

```
BEGIN QUESTION
name: q2_1_1
```

```
In [32]: probability_shiny_given_large = 4/5 #SOLUTION
```

```
In [33]: # TEST
         np.isclose(probability_shiny_given_large, 4/5)
```

```
Out[33]: True
```

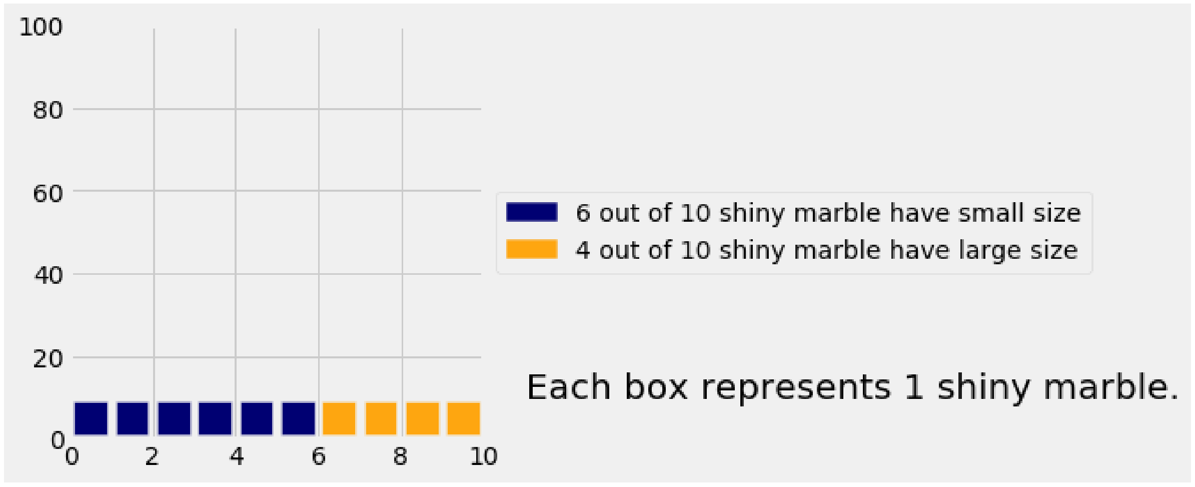
You should have found that this is different from the probability that you're a shiny marble given no size information) which you computed earlier. The distribution of surfaces among the large marbles is a little different from the distribution of surfaces among all the marbles.

Question 2.1.2. Suppose instead Samantha had said you're a **shiny** marble (hooray!). What's the probability that you're large?

Run the code cell below to display the icon array, then assign `probability_large_given_shiny` to the appropriate value.

```
BEGIN QUESTION
name: q2_1_2
```

```
In [34]: # Run this cell to display the icon array. Then fill in the last line.
         display_grouped_icon_array(marbles.where("surface", "shiny").group("si
         ze"), "shiny marble")

         # Now fill in the answer.
         probability_large_given_shiny = 4/10 #SOLUTION
```

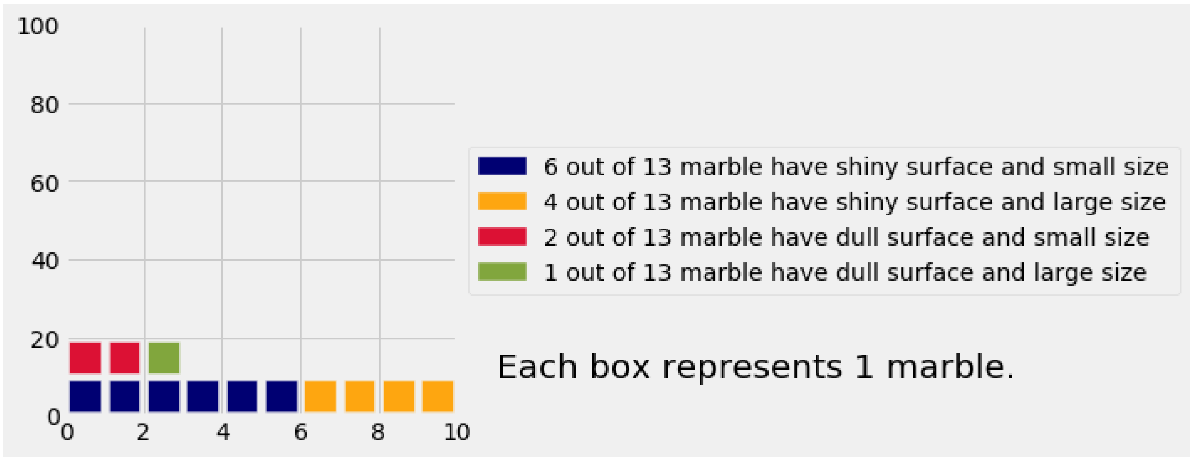


```
In [35]: # TEST
         np.isclose(probability_large_given_shiny, 4/10)
```

```
Out[35]: True
```

Question 2.1.3. Can you answer the previous two questions just by looking at the full icon array? (You can run the cell below to see it again.).

In [36]:
```
# Just run this cell.  The next cell is where you should write your an
swer.
display_grouped_icon_array(marbles.group(["surface", "size"]), "marbl
e")
```



If you can, how? If not, why not? Check with your lab peers to see if you are on the right track.

```
BEGIN QUESTION
name: q2_1_3
```

SOLUTION: To answer question 2.1.2, we can simply look at the shiny marbles. There are 10 total, and 6 of those are large. To answer question 2.1.1, we can simply look only at the large marbles. There are 5 total, and 4 of them are shiny.

3. Cancer screening

Hopefully the icon arrays from the previous portion helped you build intuition for why conditional probabilities can be helpful. Now, let’s look at a real life application.

Background

Medical tests are an important but surprisingly controversial topic. For years, women have been advised to get regular mammograms, which test for breast cancer. Today, there is controversy over whether the tests are useful at all.

Part of the problem with such tests is that they are not perfectly reliable. Someone without cancer, or with only a benign form of cancer, can see a positive result on a test for cancer. Someone with cancer can receive a negative result. ("Positive" means "pointing toward cancer," so in this context it's bad!) Doctors and patients often deal poorly with the first case, called *false positives*. For example, a patient may receive dangerous treatment like chemotherapy or radiation despite having no cancer or, as happens more frequently, having a cancer that would not have impacted their health.

Conditional probability is a good way to think about such situations. For example, you can compute the chance that you have cancer **given the results of a diagnostic test** by combining information from different probability distributions. You'll see that the chance you have cancer can be far from 100% even if you have a positive test result from a test that is usually accurate.

3.1. Basic cancer statistics

Suppose that in a representative group of 10,000 people who are tested for cancer ("representative" meaning that the frequencies of different events are the same as the frequencies in the whole population):

1. 100 people have cancer.
2. Among the 100 people that have cancer, 90 have positive results on a cancer test and 10 have negative results. ("Negative" means "not pointing toward cancer.")
3. The other 9,900 people don't have cancer.
4. Among these 9,900 people, 198 have positive results on a cancer test and the other 9,702 have negative results. (So 198 see "false positive" results.)

Below we've generated a table with data from these 10,000 hypothetical people.

In [37]:
```
people = Table().with_columns(
    "cancer status", ["sick", "sick", "healthy", "healthy"],
    "test status", ["positive", "negative", "positive", "negative"],
    "count", [90, 10, 198, 9702])
people
```

Out[37]:

cancer status	test status	count
sick	positive	90
sick	negative	10
healthy	positive	198
healthy	negative	9702

One way to visualize this dataset is with a contingency table, which you've seen before.

Question 3.1.1. Create a contingency table that looks like this:

cancer status	negative	positive
healthy		
sick		

...with the **count** of each group filled in, according to what we've told you above. The counts in the 4 boxes should sum to 10,000.

Hint: Use `pivot` with the `sum` function.

```
BEGIN QUESTION
name: q3_1_1
```

In [38]:
```
cancer = people.pivot("test status", "cancer status", "count", sum) #S
OLUTION
cancer
```

Out[38]:

cancer status	negative	positive
healthy	9702	198
sick	10	90

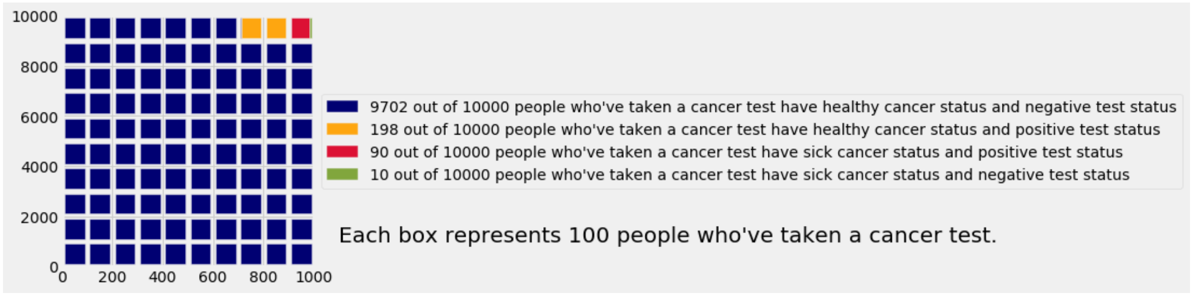
In [39]:
```
# TEST
cancer
```

Out[39]:

cancer status	negative	positive
healthy	9702	198
sick	10	90

Here is the `people` data in an icon array.

In [40]:
```
display_grouped_icon_array(people, "people who've taken a cancer test"
)
```

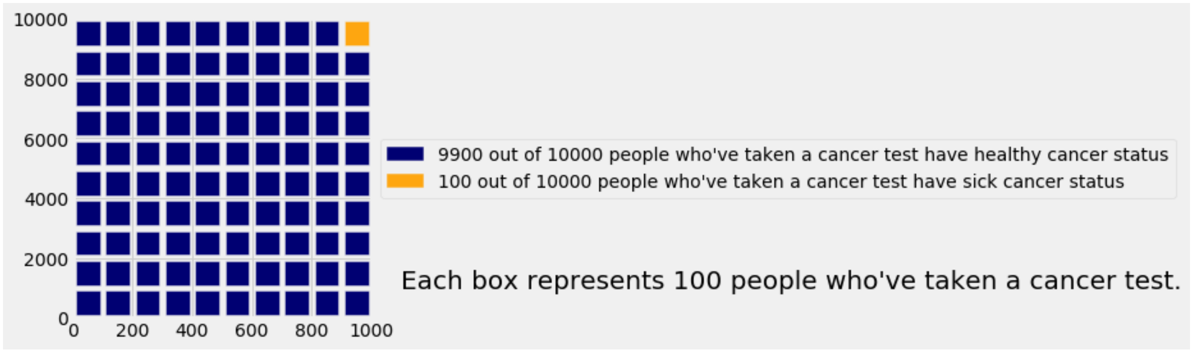


Now let's think about how you can use this kind of information when you're tested for cancer.

Before you know any information about yourself, you could imagine yourself as a **uniform random sample** of one of the 10,000 people in this imaginary population of people who have been tested.

What's the chance that you have cancer, knowing nothing else about yourself? It's $\frac{100}{10000}$, or 1%. We can see that more directly with this icon array:

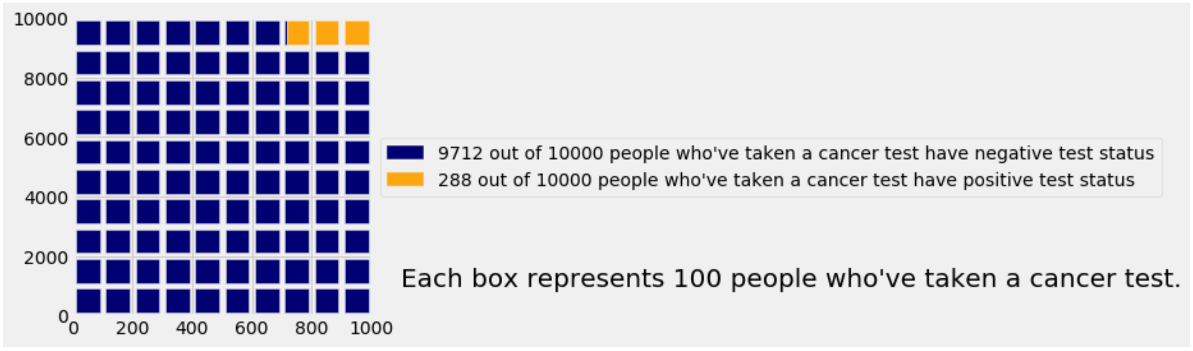
In [41]:
```
by_health = people.select(0, 2).group(0, sum).relabeled(1, 'count')
display_grouped_icon_array(by_health, "people who've taken a cancer te
st")
```



Question 3.1.2. What's the chance that you have a positive test result, knowing nothing else about yourself? Run the next code cell to dispaly an icon array, then assign `probability_positive_test` to this value.

```
BEGIN QUESTION
name: q3_1_2
```

In [42]:
```
# Run this cell to display an icon array. Then fill in the probability
of a positive test result in the last line.
by_test = people.select(1, 2).group(0, sum).relabeled(1, 'count')
display_grouped_icon_array(by_test, "people who've taken a cancer tes
t")

# Now fill in the probability of a positive test result
probability_positive_test = 288/10000 # SOLUTION
```



In [43]:
```
# TEST
np.isclose(probability_positive_test, 288/10000)
```

Out[43]:
```
True
```

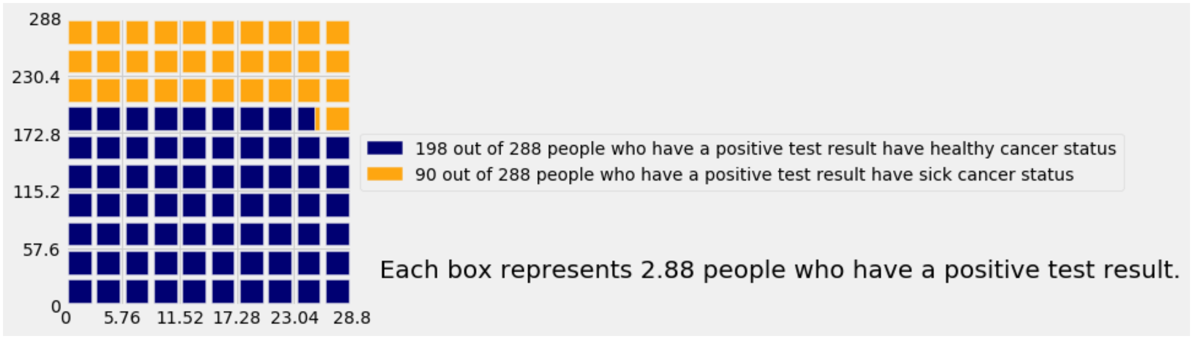
3.2. Interpreting test results

Suppose you have a positive test result. This means that you can now narrow yourself down to being part of just one of the two following groups:

1. The people with cancer who have a positive test result.
2. The people without cancer who have a positive test result.

Here's an icon array for those two groups:

In [44]:
```
# Just run this cell.
display_grouped_icon_array(people.where("test status", are.equal_to("p
ositive")).drop(1), "people who have a positive test result")
```



The *conditional probability* that you **have cancer given your positive test result** is the chance that you're in the first group (cancer), assuming you have a positive test result.

Question 3.2.1. Eyeballing the icon array above, is the conditional probability that you have cancer **given your positive test result** closest to:

1. 9/10
2. 2/3
3. 1/2
4. 1/3
5. 1/100

Assign `rough_prob_sick_given_positive` to the number corresponding to your answer.

```
BEGIN QUESTION
name: q3_2_1
```

In [45]:
```
# Set this to one of the probabilities above.
rough_prob_sick_given_positive = 4 #SOLUTION
```

In [46]:
```
# TEST
rough_prob_sick_given_positive == 4
```

Out[46]:
```
True
```

Question 3.2.2. Now write code to calculate that probability exactly, using the original contingency table you wrote (the `cancer` table).

Run the next code cell to see the `cancer` table, then fill in `prob_sick_given_positive` with your code.

```
BEGIN QUESTION
name: q3_2_2
```

In [47]:
```
prob_sick_given_positive = cancer.where("cancer status", are.equal_to(
"sick")).column("positive").item(0) / sum(cancer.column("positive")) #
SOLUTION
prob_sick_given_positive

# Run this cell first to see the cancer table. Then fill in the first
 line of this cell.
print(cancer)

print('Probability of cancer given positive test result: {}'.format(pr
ob_sick_given_positive))
```

```
cancer status | negative | positive
healthy       | 9702     | 198
sick          | 10       | 90
Probability of cancer given positive test result: 0.3125
```

```
In [48]: # TEST
         np.isclose(prob_sick_given_positive, 0.3125)

Out[48]: True
```

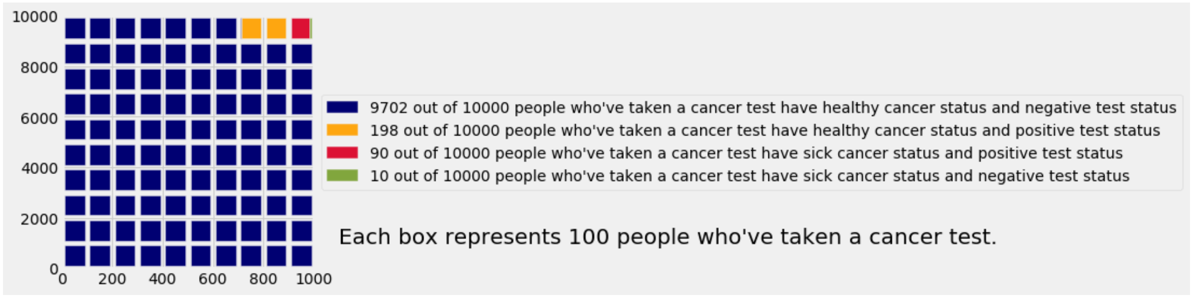
Question 3.2.3. Look at the full icon array again. Using that, how would you compute the conditional probability of cancer given a positive test?

Run the next code cell to see the full icon array.

```
BEGIN QUESTION
name: q3_2_3
```

SOLUTION: Look only at the boxes for positive test statuses (yellow and red). Among those, 90/288, or 31.25%, have cancer.

```
In [49]: # The full icon array is given here for your convenience.
         # Write your answer in the previous cell.
         display_grouped_icon_array(people, "people who've taken a cancer test"
         )
```



Question 3.2.4. Is your answer to question 3.2.2 bigger than the overall proportion of people in the population who have cancer (given as 1% in 3.1.1)? Does that make sense? Check with your peers in lab to see if you have the right idea.

```
BEGIN QUESTION
name: q3_2_4
```

SOLUTION: Yes. A positive test means your chance of having cancer is higher. Otherwise the test wouldn't really be a cancer test at all!

4. Tree diagrams

A tree diagram is another useful visualization that helps us calculate conditional probabilities. It is easiest to draw a tree diagram when the probabilities are presented in a slightly different way - specifically, we first look at the probabilities of cancer in the entire population, and then examine the rates of positive test results within the cancer and no cancer groups. For example, people often summarize the information in your `cancer` table using 3 numbers:

1. The overall probability of having cancer is **`p_cancer`**. (This is called the *base rate* or *marginal probability* of the disease.)
2. Given that you have cancer, the probability of a positive test result is **`p_pos_given_cancer`**. (This is called the *sensitivity* of the test. Higher values of `p_pos_given_cancer` mean the test is more useful.)
3. Given that you don't have cancer, the probability of a positive test result is **`p_pos_given_nocancer`**. (This is called the *false positive rate* of the test. Higher values of `p_pos_given_nocancer` mean the test is less useful.)

You already saw that the base rate of cancer was .01 in the previous section. `p_pos_given_cancer` and `p_pos_given_nocancer` can be computed using the same method you used to compute the conditional probability of cancer given a positive test result.

The information we have on cancer can be represented in this tree diagram:

```
                  /    \
             .99 /      \ .01
                /        \
               /          \
          no cancer      cancer
            / \            / \
       0.1 /   \ 0.9  0.98/   \ 0.02
          /     \        /     \
         +       -      +       -
```

Question 4.1. Compute `p_pos_given_cancer` and `p_pos_given_nocancer` for the data in section 3.

Hint: Use Bayes' Rule. For your reference, chapter 18.1 of the textbook has a section on [tree diagrams (https://www.inferentialthinking.com/chapters/18/1/More_Likely_than_Not_Binary_Classifier.html#tree-diagram)](https://www.inferentialthinking.com/chapters/18/1/More_Likely_than_Not_Binary_Classifier.html#tree-diagram) and [Bayes' Rule (https://www.inferentialthinking.com/chapters/18/1/More_Likely_than_Not_Binary_Classifier.html#bayes-rule)](https://www.inferentialthinking.com/chapters/18/1/More_Likely_than_Not_Binary_Classifier.html#bayes-rule).

```
BEGIN QUESTION
name: q4_1
```

In [50]:
```
# Hint: You may find these two tables useful:
has_cancer = cancer.where("cancer status", are.equal_to("sick"))
no_cancer = cancer.where("cancer status", are.equal_to("healthy"))

p_cancer = .01
p_pos_given_cancer = (has_cancer.column("positive").item(0) / 10000) /
p_cancer #SOLUTION
p_pos_given_nocancer = (no_cancer.column("positive").item(0) / 10000)
/ (1 - p_cancer) #SOLUTION

print('Probability of Cancer:', p_cancer, '\nProbability of a positive
test given cancer:', p_pos_given_cancer,
      '\nProbability of a positive test given no cancer:', p_pos_given
_nocancer)
```

```
Probability of Cancer: 0.01
Probability of a positive test given cancer: 0.8999999999999999
Probability of a positive test given no cancer: 0.02
```

In [51]:
```
# TEST
np.isclose(p_pos_given_cancer, 0.9)
```

Out[51]:
```
True
```

In [52]:
```
# TEST
np.isclose(p_pos_given_nocancer, 0.02)
```

Out[52]:
```
True
```

Question 4.2. What is the difference in assumptions between the first probability in question 4.1 (probability of a positive test given cancer) and the one in question 3.2.2 (probability of cancer given a positive test result)?

What kind of information does each probability tell us?

```
BEGIN QUESTION
name: q4_2
```

SOLUTION: In 3.2.2, we were given the information that we had a positive test result, and then calculated the chance of having cancer. In question 4.1, we started out with the assumption that we knew the individual had cancer, and then calculated the chance of a positive test result. The probability in question 4.1 tells us how useful the test is - if it was lower than the probability of a positive test result given *no* cancer, we'd probably be better off not using the test! The probability in question 3.2.2 tells us that given a positive test result, an individual is more likely to *not* have cancer. It highlights the uncertainty in an initial cancer test and the need for further testing to confirm (or disprove) the preliminary results.

Congratulations, you're done with the final lab! Be sure to

- **run all the tests** (the next cell has a shortcut for that),
- **Save and Checkpoint** from the `File` menu,
- **run the last cell to submit your work**,
- and ask one of the staff members to check you off.

In [58]:
```
# For your convenience, you can run this cell to run all the tests at once!
import os
_ = [ok.grade(q[:-3]) for q in os.listdir("tests") if q.startswith('q')]
```

In [59]:
```
_ = ok.submit()
```

```
Saving notebook... No valid file sources found
Submit... 0.0% complete
Could not submit: Assignment does not exist
Backup... 0.0% complete
Could not backup: Assignment does not exist
```