This notebook is an exercise in the Intermediate Machine Learning course. You can reference the tutorial at this link.

Most people find target leakage very tricky until they've thought about it for a long time.

So, before trying to think about leakage in the housing price example, we'll go through a few examples in other applications. Things will feel more familiar once you come back to a question about house prices.

Setup

The questions below will give you feedback on your answers. Run the following cell to set up the feedback system.

In [1]:
```
# Set up code checking
from learntools.core import binder
binder.bind(globals())
from learntools.ml_intermediate.ex7 import *
print("Setup Complete")
```

```
Setup Complete
```

Step 1: The Data Science of Shoelaces

Nike has hired you as a data science consultant to help them save money on shoe materials. Your first assignment is to review a model one of their employees built to predict how many shoelaces they'll need each month. The features going into the machine learning model include:

- The current month (January, February, etc)
- Advertising expenditures in the previous month
- Various macroeconomic features (like the unemployment rate) as of the beginning of the current month
- The amount of leather they ended up using in the current month

The results show the model is almost perfectly accurate if you include the feature about how much leather they used. But it is only moderately accurate if you leave that feature out. You realize this is because the amount of leather they use is a perfect indicator of how many shoes they produce, which in turn tells you how many shoelaces they need.

Do you think the *leather used* feature constitutes a source of data leakage? If your answer is "it depends," what does it depend on?

After you have thought about your answer, check it against the solution below.

In [2]:
```
# Check your answer (Run this code cell to receive credit!)
q_1.check()
```

Correct:

This is tricky, and it depends on details of how data is collected (which is common when thinking about leakage). Would you at the beginning of the month decide how much leather will be used that month? If so, this is ok. But if that is determined during the month, you would not have access to it when you make the prediction. If you have a guess at the beginning of the month, and it is subsequently changed during the month, the actual amount used during the month cannot be used as a feature (because it causes leakage).

Step 2: Return of the Shoelaces

You have a new idea. You could use the amount of leather Nike ordered (rather than the amount they actually used) leading up to a given month as a predictor in your shoelace model.

Does this change your answer about whether there is a leakage problem? If you answer "it depends," what does it depend on?

In [3]:
```
# Check your answer (Run this code cell to receive credit!)
q_2.check()
```

Correct:

This could be fine, but it depends on whether they order shoelaces first or leather first. If they order shoelaces first, you won't know how much leather they've ordered when you predict their shoelace needs. If they order leather first, then you'll have that number available when you place your shoelace order, and you should be ok.

Step 3: Getting Rich With Cryptocurrencies?

You saved Nike so much money that they gave you a bonus. Congratulations.

Your friend, who is also a data scientist, says he has built a model that will let you turn your bonus into millions of dollars. Specifically, his model predicts the price of a new cryptocurrency (like Bitcoin, but a newer one) one day ahead of the moment of prediction. His plan is to purchase the cryptocurrency whenever the model says the price of the currency (in dollars) is about to go up.

The most important features in his model are:

- Current price of the currency
- Amount of the currency sold in the last 24 hours
- Change in the currency price in the last 24 hours

- Change in the currency price in the last 1 hour
- Number of new tweets in the last 24 hours that mention the currency

The value of the cryptocurrency in dollars has fluctuated up and down by over $100 in the last year, and yet his model's average error is less than $1. He says this is proof his model is accurate, and you should invest with him, buying the currency whenever the model says it is about to go up.

Is he right? If there is a problem with his model, what is it?

In [4]:
```
# Check your answer (Run this code cell to receive credit!)
q_3.check()
```

Correct:

There is no source of leakage here. These features should be available at the moment you want to make a predition, and they're unlikely to be changed in the training data after the prediction target is determined. But, the way he describes accuracy could be misleading if you aren't careful. If the price moves gradually, today's price will be an accurate predictor of tomorrow's price, but it may not tell you whether it's a good time to invest. For instance, if it is $100 today, a model predicting a price of $100 tomorrow may seem accurate, even if it can't tell you whether the price is going up or down from the current price. A better prediction target would be the change in price over the next day. If you can consistently predict whether the price is about to go up or down (and by how much), you may have a winning investment opportunity.

Step 4: Preventing Infections

An agency that provides healthcare wants to predict which patients from a rare surgery are at risk of infection, so it can alert the nurses to be especially careful when following up with those patients.

You want to build a model. Each row in the modeling dataset will be a single patient who received the surgery, and the prediction target will be whether they got an infection.

Some surgeons may do the procedure in a manner that raises or lowers the risk of infection. But how can you best incorporate the surgeon information into the model?

You have a clever idea.

1. Take all surgeries by each surgeon and calculate the infection rate among those surgeons.
2. For each patient in the data, find out who the surgeon was and plug in that surgeon's average infection rate as a feature.

Does this pose any target leakage issues? Does it pose any train-test contamination issues?

In [5]:
```
# Check your answer (Run this code cell to receive credit!)
q_4.check()
```

Correct:

This poses a risk of both target leakage and train-test contamination (though you may be able to avoid both if you are careful).

You have target leakage if a given patient's outcome contributes to the infection rate for his surgeon, which is then plugged back into the prediction model for whether that patient becomes infected. You can avoid target leakage if you calculate the surgeon's infection rate by using only the surgeries before the patient we are predicting for. Calculating this for each surgery in your training data may be a little tricky.

You also have a train-test contamination problem if you calculate this using all surgeries a surgeon performed, including those from the test-set. The result would be that your model could look very accurate on the test set, even if it wouldn't generalize well to new patients after the model is deployed. This would happen because the surgeon-risk feature accounts for data in the test set. Test sets exist to estimate how the model will do when seeing new data. So this contamination defeats the purpose of the test set.

Step 5: Housing Prices

You will build a model to predict housing prices. The model will be deployed on an ongoing basis, to predict the price of a new house when a description is added to a website. Here are four features that could be used as predictors.

1. Size of the house (in square meters)
2. Average sales price of homes in the same neighborhood
3. Latitude and longitude of the house
4. Whether the house has a basement

You have historic data to train and validate the model.

Which of the features is most likely to be a source of leakage?

In [6]:
```
# Fill in the line below with one of 1, 2, 3 or 4.
potential_leakage_feature = 2

# Check your answer
q_5.check()
```

Correct:

2 is the source of target leakage. Here is an analysis for each feature:

1. The size of a house is unlikely to be changed after it is sold (though technically it's possible). But typically this will be available when we need to make a prediction, and the data won't be modified after the home is sold. So it is pretty safe.

2. We don't know the rules for when this is updated. If the field is updated in the raw data after a home was sold, and the home's sale is used to calculate the average, this constitutes a case of target leakage. At an extreme, if only one home is sold in the neighborhood, and it is the home we are trying to predict, then the average will be exactly equal to the value we are trying to predict. In general, for neighborhoods with few sales, the model will perform very well on the training data. But when you apply the model, the home you are predicting won't have been sold yet, so this feature won't work the same as it did in the training data.

3. These don't change, and will be available at the time we want to make a prediction. So there's no risk of target leakage here.

4. This also doesn't change, and it is available at the time we want to make a prediction. So there's no risk of target leakage here.

In [7]:
```
#q_5.hint()
#q_5.solution()
```

Conclusion

Leakage is a hard and subtle issue. You should be proud if you picked up on the issues in these examples.

Now you have the tools to make highly accurate models, and pick up on the most difficult practical problems that arise with applying these models to solve real problems.

There is still a lot of room to build knowledge and experience. Try out a Competition or look through our Datasets to practice your new skills.

Again, Congratulations!

Have questions or comments? Visit the course discussion forum to chat with other learners.