Traitement de l'overfitting

Later in this specialization, we'll talk about debugging and diagnosing things that can go wrong with learning algorithms. You'll also learn about specific tools to recognize when overfitting and underfitting may be occurring. But for now, when you think overfitting has occurred, lets talk about what you can do to address it.

Let's say you fit a model and it has high variance, is overfit. Here's our overfit house price prediction model. One way to address this problem is to collect more training data, that's one option. If you're able to get more data, that is more training examples on sizes and prices of houses, then with the larger training set, the learning algorithm will learn to fit a function that is less wiggly.

You can continue to fit a high order polynomial or some of the function with a lot of features, and if you have enough training examples, it will still do okay. To summarize, the number one tool you can use against overfitting is to get more training data. Now, getting more data isn't always an option. Maybe only so many houses have been sold in this location, so maybe there just isn't more data to be add. But when the data is available, this can work really well.

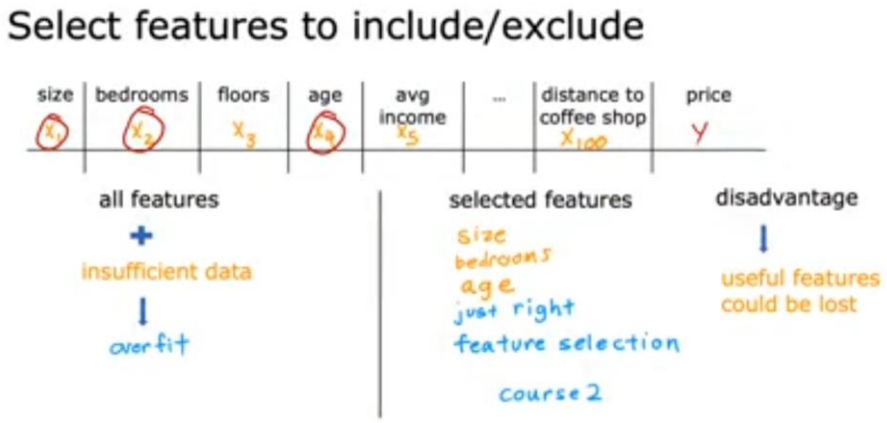
A second option for addressing overfitting is to see if you can use fewer features. In the previous video, our models features included the size x, as well as the size squared, and this x squared, and x cubed and x^4 and so on. These were a lot of polynomial features. In that case, one way to reduce overfitting is to just not use so many of these polynomial features.

But now let's look at a different example. Maybe you have a lot of different features of a house of which to try to predict its price, ranging from the size, number of bedrooms, number of floors, the age, average income of the neighborhood, and so on and so forth, total distance to the nearest coffee shop. It turns out that if you have a lot of features like these but don't have enough training data, then your learning algorithm may also overfit to your training set.

Now instead of using all 100 features, if we were to pick just a subset of the most useful ones, maybe size, bedrooms, and the age of the house. If you think those are the most relevant features, then using just that smallest subset of features, you may find that your model no longer overfits as badly. Choosing the most appropriate set of features to use is sometimes also called feature selection.

One way you could do so is to use your intuition to choose what you think is the best set of features, what's most relevant for predicting the price. Now, one disadvantage of feature selection is that by using only a subset of the features, the algorithm is throwing away some of the information that you have about the houses.

For example, maybe all of these features, all 100 of them are actually useful for predicting the price of a house. Maybe you don't want to throw away some of the information by throwing away some of the features. Later in Course 2, you'll also see some algorithms for automatically choosing the most appropriate set of features to use for our prediction task.

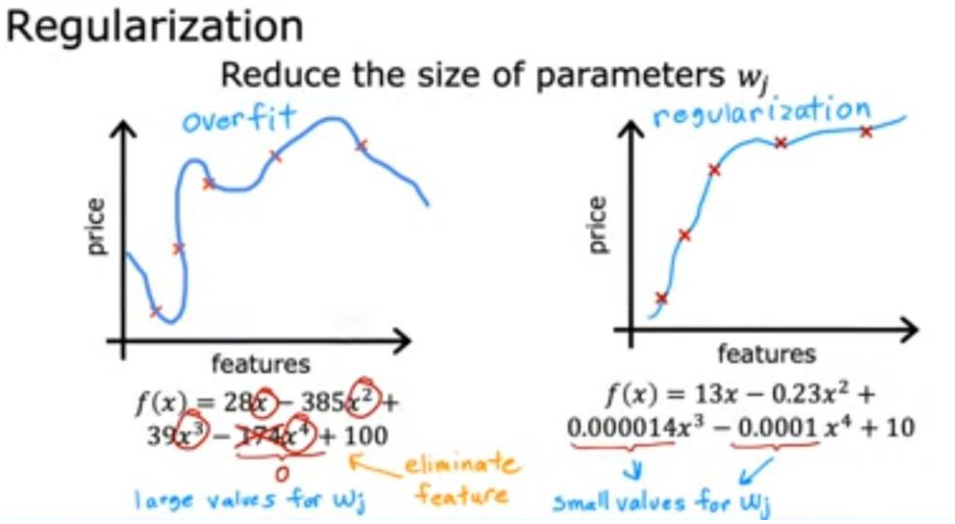


Now, this takes us to the third option for reducing overfitting. This technique, which we'll look at in even greater depth in the next video is called regularization. If you look at an overfit model, here's a model using polynomial features: x, x squared, x cubed, and so on. You find that the parameters are often relatively large.

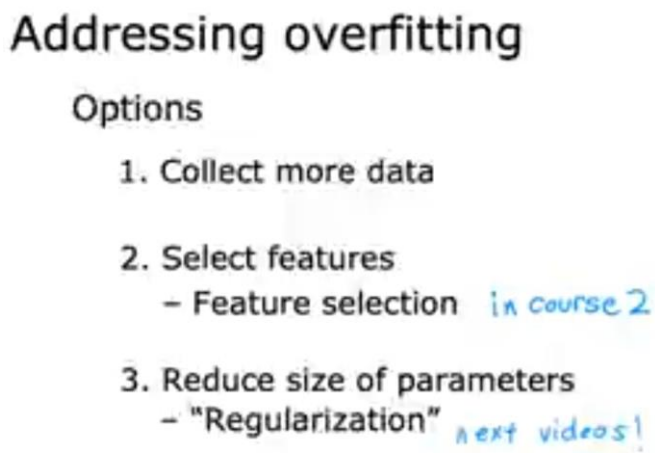
Now if you were to eliminate some of these features, say, if you were to eliminate the feature x4, that corresponds to setting this parameter to 0. So setting a parameter to 0 is equivalent to eliminating a feature, which is what we saw on the previous slide. It turns out that regularization is a way to more gently reduce the impacts of some of the features without doing something as harsh as eliminating it outright.

What regularization does is encourage the learning algorithm to shrink the values of the parameters without necessarily demanding that the parameter is set to exactly 0. It turns out that even if you fit a higher order polynomial like this, so long as you can get the algorithm to use smaller parameter values: w1, w2, w3, w4.

You end up with a curve that ends up fitting the training data much better. So what regularization does, is it lets you keep all of your features, but they just prevents the features from having an overly large effect, which is what sometimes can cause overfitting.



By the way, by convention, we normally just reduce the size of the wj parameters, that is w1 through wn. It doesn't make a huge difference whether you regularize the parameter b as well, you could do so if you want or not if you don't. I usually don't and it's just fine to regularize w1, w2, all the way to wn, but not really encourage b to become smaller. In practice, it should make very little difference whether you also regularize b or not.



To recap, these are the three ways you saw in this video for addressing overfitting. One, collect more data. If you can get more data, this can really help reduce overfitting. Sometimes that's not possible. In which case, some of the options are, two, try selecting and using only a subset of the features. You'll learn more about feature selection in Course 2.

Three would be to reduce the size of the parameters using regularization. This will be the subject of the next video as well. Just for myself, I use regularization all the time. So this is a very useful technique for training learning algorithms, including neural networks specifically, which you'll see later in this specialization as well.