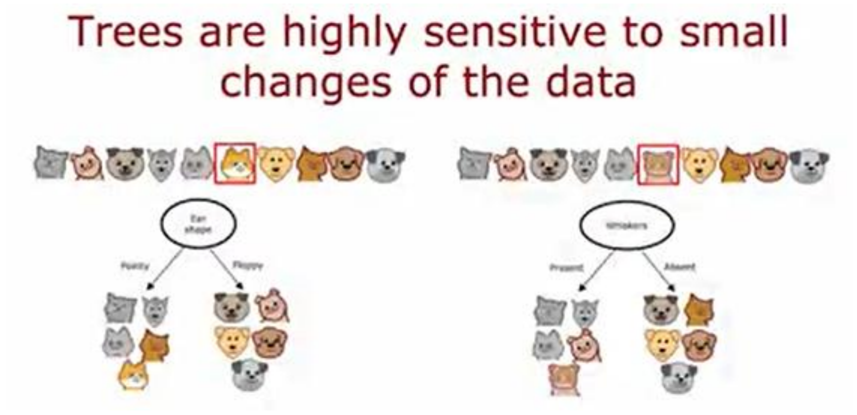
Using multiple decision trees

One of the weaknesses of using a single decision tree is that that decision tree can be highly sensitive to small changes in the data. One solution to make the algorithm less sensitive or more robust is to build not one decision tree, but to build a lot of decision trees, and we call that a tree together. Let's take a look.

With the example that we've been using, the best feature to split on at the root node turned out to be the ear shape resulting in these two subsets of the data and then building further sub trees on these two subsets of the data. But it turns out that if you were to take just one of the ten examples and change it to a different cat so that instead of having pointy ears, round face, whiskers absent, this new cat has floppy ears, round face, whiskers present, with just changing a single training example, the highest information gain feature to split on becomes the whiskers feature instead of the ear shape feature.

As a result of that, the subsets of data you get in the left and right sub-trees become totally different and as you continue to run the decision tree learning algorithm recursively, you build out totally different sub trees on the left and right. The fact that changing just one training example causes the algorithm to come up with a different split at the root and therefore a totally different tree, that makes this algorithm just not that robust.



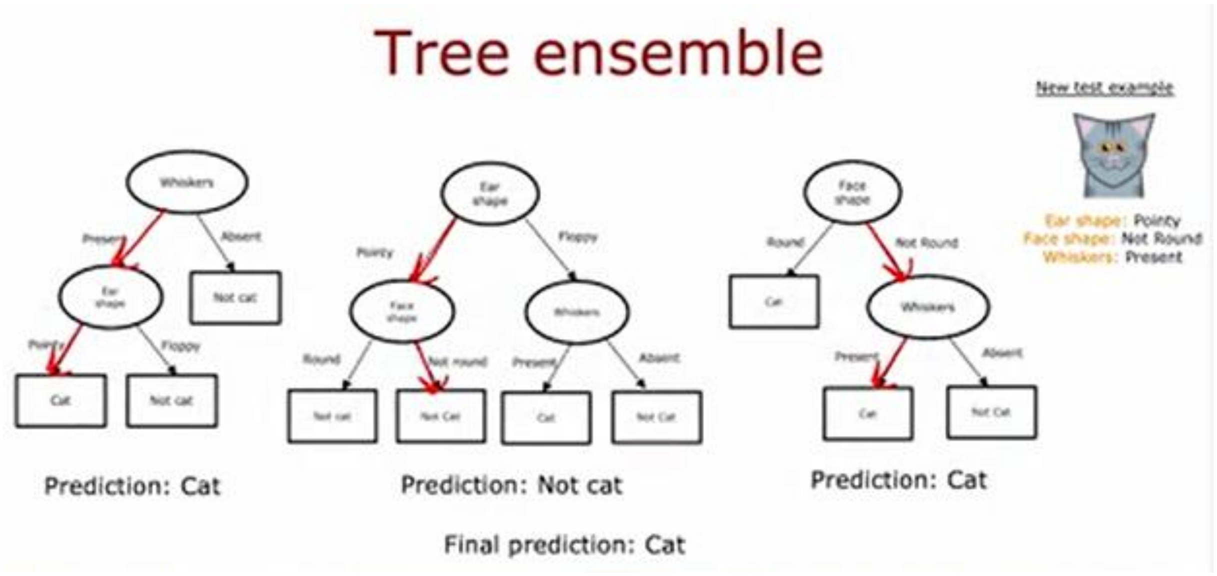
That's why when you're using decision trees, you often get a much better result, that is, you get more accurate predictions if you train not just a single decision tree but a whole bunch of different decision trees. This is what we call a tree ensemble, which just means a collection of multiple trees. We'll see, in the next few videos, how to construct this ensemble of trees. But if you had this ensemble three trees, each one of these is maybe a plausible way to classify cat versus not cat.

If you had a new test example that you wanted to classify, then what you would do is run all three of these trees on your new example and get them to vote on whether it's the final prediction. This test example has pointy ears, a not round face shape and whiskers are present and so the first tree would carry out inferences like this and predict that it is a cat. The second tree's inference would follow this path through the tree and therefore predict that is not cat.

The third tree would follow this path and therefore predict that it is a cat. These three trees have made different predictions and so what we'll do is actually get them to vote. The majority votes of the predictions among these three trees is, cat. The final prediction of this ensemble of trees is that this is a cat which happens to be the correct prediction.

The reason we use an ensemble of trees is by having lots of decision trees and having them vote, it makes your overall algorithm less sensitive to what any single tree may be doing because it gets only one vote out of three or one vote out of many, many different votes and it makes your overall algorithm more robust. But how do you come up with all of these different plausible but maybe slightly different decision trees in order to get them to vote?

In the next video, we'll talk about a technique from statistics called sampling with replacement and this will turn out to be a key technique that we'll use in the video after that in order to build this ensemble of trees.


Tree ensemble
New test example
Ear shape: Pointy
Face shape: Not Round
Whiskers: Present
Whiskers
Present
Absent
Ear shape
Not cat
Pointy
Floppy
Cat
Not cat
Prediction: Cat
Ear shape
Pointy
Floppy
Face shape
Whiskers
Round
Not round
Present
Absent
Not cat
Not Cat
Cat
Not Cat
Prediction: Not cat
Face shape
Round
Not Round
Cat
Whiskers
Present
Absent
Cat
Not Cat
Prediction: Cat
Final prediction: Cat