Cracteristics of continuous valueha

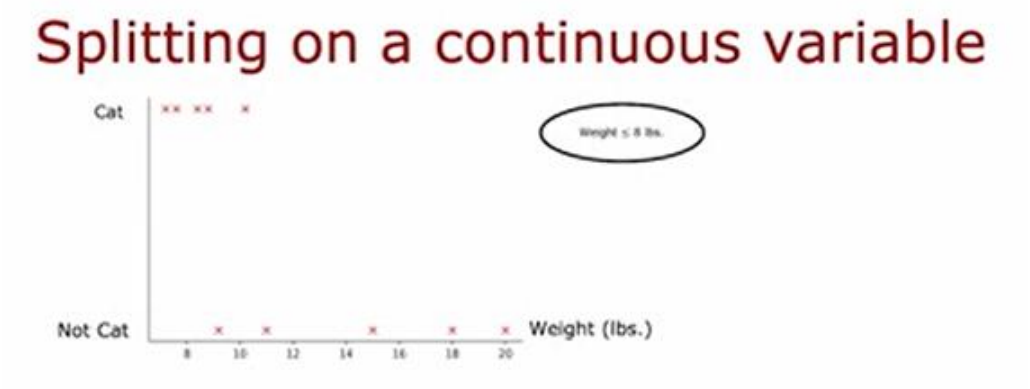
Let's look at how you can modify decision tree to work with features that aren't just discrete values but continuous values. That is features that can be any number. Let's start with an example, I have modified the cat adoption center of data set to add one more feature which is the weight of the animal. In pounds on average between cats and dogs, cats are a little bit lighter than dogs, although there are some cats are heavier than some dogs.

But so the weight of an animal is a useful feature for deciding if it is a cat or not. So how do you get a decision tree to use a feature like this? The decision tree learning algorithm will proceed similarly as before except that rather than constraint splitting just on ear shape, face shape and whiskers.

You have to consist splitting on ear shape, face shape whisker or weight. And if splitting on the weight feature gives better information gain than the other options. Then you will split on the weight feature. But how do you decide how to split on the weight feature? Let's take a look. Here's a plot of the data at the root. Not plotted on the horizontal axis.

The way to the animal and the vertical axis is cat on top and not cat below. So the vertical axis indicates the label, y being 1 or 0. The way we were split on the weight feature would be if we were to split the data based on whether or not the weight is less than or equal to some value. Let's say 8 or some of the number.

That will be the job of the learning algorithm to choose. And what we should do when constraints splitting on the weight feature is to consider many different values of this threshold and then to pick the one that is the best. And by the best I mean the one that results in the best information gain.



So in particular, if you were considering splitting the examples based on whether the weight is less than or equal to 8, then you will be splitting this data set into two subsets. Where the subset on the

left has two cats and the subset on the right has three cats and five dogs. So if you were to calculate our usual information gain calculation, you'll be computing the entropy at the root node NC p f 0.5 minus now 2/10 times entropy of the left split has two other two cats.

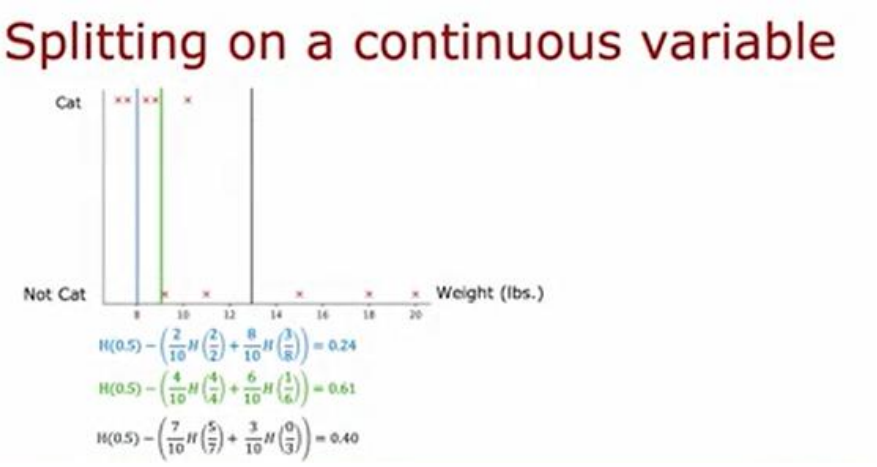
So it should be 2/2 plus the right split has eight out of 10 examples and an entropy F. That's of the eight examples on the right three cats. To entry of 3/8 and this turns out to be 0.24. So this would be information gain if you were to split on whether the weight is less than equal to 8 but we should try other values as well.

So what if you were to split on whether or not the weight is less than equal to 9 and that corresponds to this new line over here. And the information gain calculation becomes H (0.5) minus. So now we have four examples and left split all cats. So that's 4/10 times entropy of 4/4 plus six examples on the right of which you have one cat.

So that's 6/10 times each of 1/6, which is equal to turns out 0.61. So the information gain here looks much better is 0.61 information gain which is much higher than 0.24. Or we could try another value say 13.

And the calculation turns out to look like this, which is 0.40. In the more general case, we'll actually try not just three values, but multiple values along the X axis. And one convention would be to sort all of the examples according to the weight or according to the value of this feature and take all the values that are mid points between the sorted list of training.

Examples as the values for consideration for this threshold over here. This way, if you have 10 training examples, you will test nine different possible values for this threshold and then try to pick the one that gives you the highest information gain. And finally, if the information gained from splitting on a given value of this threshold is better than the information gain from splitting on any other feature, then you will decide to split that node at that feature. And in this example an information gain of 0.61 turns out to be higher than that of any other feature.



It turns out they're actually two thresholds. And so assuming the algorithm chooses this feature to split on, you will end up splitting the data set according to whether or not the weight of the animal is less than equal to £9. And so you end up with two subsets of the data like this and you can then build recursively, additional decision trees using these two subsets of the data to build out the rest of the tree. So to summarize to get the decision tree to work on continuous value features at every note.

When consuming splits, you would just consider different values to split on, carry out the usual information gain calculation and decide to split on that continuous value feature if it gives the highest possible information gain. So that's how you get the decision tree to work with continuous value features.

Try different thresholds, do the usual information gain calculation and split on the continuous value feature with the selected threshold if it gives you the best possible information gain out of all possible features to split on.

And that's it for the required videos on the core decision tree algorithm After there's there is an optional video you can watch or not that generalizes the decision tree learning algorithm to regression trees. So far, we've only talked about using decision trees to make predictions that are classifications predicting a discrete category, such as cat or not cat. But what if you have a regression problem where you want to predict a number in the next video. I'll talk about a generalization of decision trees to handle that.