Hello guys. So we are going to continue the discussion with respect to machine learning. And in this video we are going to start a new machine learning algorithm which is called as k nearest neighbor. So out of all the machine learning algorithm we have learnt till now, k nearest neighbor is the most easiest one because you'll be able to understand it, you know, and with the help of both K uh, with the help of k nearest neighbor, we can solve both the problem that is classification and regression problem statement. Now let's go ahead and let's try to understand what exactly we are doing with respect to the k nearest neighbor, what is the mathematical intuition behind it and what mathematical formulas are also applied or used uh, with respect to this particular algorithm. So to start with, first of all I'm going to discuss about classification problem. And once you understand classification I think it will be very easy to understand about the regression problem statement. So over here in the classification with respect to the k nearest neighbor, let's consider that I have a data set okay. And uh in this particular data set let's consider that I have feature F1 f2. And this is my y features the output feature okay. Now I'll be having uh f one and f two values. And y category can be a binary category to just start with okay. It can also be a multi-class categories okay. So binary categories right now I'm just taking binary categories basically means zeros or ones. It can also have multi-class categories. Not a worries. Now with respect to F1 and F2. If I consider over here I will go ahead and plot some points. Let's say I'm plotting some points over here. And this points are basically of the category, uh, let's say zero okay. So this belongs to the category zero okay. And similarly I will go ahead and plot some other points. Okay. Some other points like this. And this category is actually belong to the category one. Okay. Now with the help of K nearest neighbor, how we are going to solve this classification problem when I probably consider this particular data set. This data set is my training data set right. Training data set okay. Now I have to probably use this particular data set or this particular algorithm to probably test my new data point. Okay. So what are the steps we specifically use. So the first step is that we have to initialize the k value. We have to initialize. The k value. Now what this k value is because the name suggests k nearest neighbor k can be any number which will be greater than zero. Okay. And it can probably go to infinity. Not a worries right. So if I say k is equal to one or k is equal to two, k is equal to three, k is equal to four, k is equal to five. That basically means one neighbor or two neighbor, or three neighbors, or four neighbors or five neighbors. Like this. We can have any value. And in order to find out the exact k value we have, this can be considered as a hyper parameter. Hyper parameter basically means this k value can change based on various data sets, right? So I will show you in the practical example how we select the k value okay. We have to definitely try different different k values and then come to a conclusion that which nearest neighbor like what should be the k value which can actually give us the highest accuracy. Okay. Now let's consider that after performing the hyper parameter tuning, I have selected the k value as five, because when k is equal to five, my accuracy of the model is very high. Okay, now in the second step, what we do is that we find the k nearest neighbor k nearest neighbor. For the test data. Now what does this basically mean? Okay. So what I'm saying is that now see let's consider that this is my training data. Now what I'm actually going to do is that let's consider that this is my new test data. Now I want to predict whether this particular data point belongs to this category or this category. Now it's very simple. First step is that we have selected what our k value is. Let's consider with the help of hyperparameter tuning we selected it I will show in the practicals like how do we select this k value okay. Now after finding out this k value this k is equal to five basically means five nearest neighbor. Now we will go ahead and find out which is the five nearest neighbor to this particular point okay. So let's say uh to this particular point I say this is my first nearest neighbor. This is my second nearest neighbor, third nearest neighbor, fourth nearest neighbor, fifth nearest neighbor. Okay. So the five nearest point I have actually found out with respect to this particular test data,

right. With respect to this particular test data point, I'm trying to find out the distance, right. Whichever is the nearest, the top five nearest right. I have selected that. Now in the third step, what we do is that we try to find out from those k values right from those k values. From those k is equal to five values. Let's say how many points, how many neighbors belong to belong to zero category zero category. And how many neighbors belong to the one category. So I'm just going ahead and counting it over here. How many zeros categories are there now with respect to zeros you will be able to see with respect to zeros okay. You will be able to see the count is one and two right. So two counts two nearest neighbor I have found out from this particular point that belongs to this particular category. Similarly with respect to ones how many points are there. 123. So three is there. Now you know that maximum number of neighbors belongs to the ones category. So your final output for this will be also one. Right. So that basically means this particular point has been now predicted as one, because the maximum number of neighbors are actually one with respect to this particular data point. This is the entire funda of the entire k nearest neighbor classification problem. But now the question arises how do I find the k nearest neighbor? For this particular test data point, we have to probably use some distance metrics, right? We have to probably use some distance formula. So what exactly is the distance formula with respect to this right. So for this we use two different types of distance formula. So the first one is something called as Manhattan distance okay. Or I will start with something called as Euclidean distance. This distance formula is used to find out the distance between two points. Right. And the second technique that we can probably apply is basically called as Manhattan distance. Okay. We'll understand what is Euclidean and one is Manhattan distance. So let's say that I have two data points over here over here. And this data point coordinate is x one comma y one. And this data point coordinate is x two comma y two. Right. In just a two dimensional space. Now if I really want to calculate the distance between this point. Right. And I hope you may be knowing this particular formula. Okay. At the end of the day, uh, we have probably learned this in linear algebra in eighth or ninth standard mathematics. Right. The distance formula with the help of Euclidean distance is just simple. That is root of x two. Minus x one plus y two minus y one. And this both will be having a whole square right. So the distance between two points or two coordinates in a two dimension will have this particular formula. In three dimension you will be having X1Y1Z1. So another one more term that I can add over here is z two minus z one. In four dimension that many number of coordinates will increase. This is a simple formula of Euclidean distance okay. And this kind of distance is used in various different applications a traffic control, right. If the flight wants to go from one place to the other place, let's say from Bangalore to New Delhi or from India to us, you know, we specifically use this Euclidean Euclidean distance concept to probably calculate the distance with respect to the air. Right. There is a second type which is called as Manhattan distance. Now in case of Manhattan distance, let's say if I have a two coordinates over here okay. And this coordinates are X1Y1. And this coordinate is X2Y2. Now in order to calculate the distance between them we will not calculate through the hypotenuse. No we will be using the base and the adjacent. So we will first of all calculate this distance. And then we will calculate this distance. When we calculate this plus this then only I will be able to get the distance between these two points. Now you may be thinking Chris where this specific distance matrix is used. Very simple guys right. If you know about uh, Iron Man movies and. All right. Or if you have seen the cities in us every cities are having basically block wise. So it is like block. So these are buildings with respect to the blocks. And in between you have roads. Right. So another distance is between another block. So there will be another another block. Now if I really want to, if a person is over here, if this person wants to go over here so he or she will book the Uber. Now Uber is going to basically take the distance which will be like this and it will be like this. Right. Not it cannot be

something like this from here to here unless and until you're traveling by helicopter. Right? So this way the Manhattan distance will be used. Now, you may be thinking, should we use Euclidean or should we use Manhattan? It depends on the problem statement. Right. But most of the time people go ahead and use with respect to Euclidean distance and uh, any one of them you just try to check whichever is the accuracy is the high. We can probably use this. Okay. Again, with respect to hyperparameter tuning we can use either one of them okay. So this way we will be able to calculate the distance between them and find the nearest point. And then we can go ahead and tell that this particular test data or new this new data point belongs to which category. Very simple very easy. Till now I hope everybody was able to understand. Now, if you have understood about classification, the second one is regression. Now regression will be super, super easy. You you don't even have to do anything. So these are your data points. Okay, so these are your data points. Okay. Let's say with respect to regression you have size and you have house price. Okay. So price size of the house and price of the house. Now let's say this will be a continuous value. Now whenever and this is your training data. Now whenever you get a new test data point let's say for this size. Right. Okay. For this size, what will be your house price? Okay. Or let's let's consider something like this for a new test data over here. Okay. What will be your house price? It's very simple. You will just go ahead and see which are the five nearest neighbors. So let's say these are my five nearest neighbor. So what you are basically going to do you are going to combine all of them, and you are basically going to find out the average of all these points to find out the output average of all this points. To find out the output. Right. That's it. In regression, all the steps are same. You have to find out the k value based on the number of neighbors. But here you will just try to find out the average. Or if you have lot of outliers you can go. Also go ahead with the median only. This is the concept entirely in the regression problem statement right now. I hope you are able to understand this. Okay, now in the next video we will also try to understand about the variance of k KNN. Okay, now it is important to understand the variance of k. Now see why I'm saying is that when I have some data points, let's say I have some two categories of data points which looks like this. This is my first category and this is my second category okay. Now whenever I get a new test data, what I have to do. Suppose, let's say this is my new test data. I have to probably go ahead and compute the nearest neighbors, right? Nearest neighbors. Don't you think this process of calculating the nearest neighbor. I have to probably calculate the distance between all these points, right? First of all, I have to calculate distance between all these points from this particular test data from this data. And then we have to take the top five nearest neighbors. Understand with respect to the time complexity. So if you have basic idea of DSA right the time complexity will be order of n. Now this will be a problem. If you have millions of data points, then calculating the distance between each and every point of the test data will be a very cumbersome process. The time complexity will obviously be high. So we use variants of KNN right. Like k d tree. So some of the trees that we specifically use is like k d tree ball tree right. So with the help of this particular tree techniques we basically optimize optimize. This entire calculation distance calculation. How do we optimize? Because we don't go ahead and check the distance between each and every points because see, understand when we say these trees right. These trees are basically created in binary tree format right now. Whenever I say binary tree, then to calculate the distance between every point to the test data right will be reduced by half. Why? Because it will not be checking with respect to every data points. And that is the power of binary tree. Right? So in the next video we'll go ahead and understand about both these variants. And we'll get to know the idea about this k-d tree and ball tree and how we are basically optimizing this particular time complexity. Because see, at the end of the day, if you really want to find a top, uh, 5 or 10 k nearest neighbors, you need to calculate the distance between every point. Why not? We just

group it in such a way that I don't have to calculate the distance between other points based on the data that I have, and that is only possible with the help of k-d tree and ball tree. And that is what I'm going to show you in the next video. So I hope you are able to understand this. We have understood this, and probably this is the most simplest algorithm in the entire machine learning. I still say it because it is very simple to understand. The only thing that you really need to remember is about Euclidean distance and Manhattan distance. That's it from my side guys. I will see you all in the next video.

Hello guys. So we are going to continue our discussion with respect to KNN and in our previous video, we have understood about the theoretical intuition. In this video we are going to discuss about the variants of KNN. Now, one problem with respect to KNN is that if I really need to find out some nearest neighbor, I need to probably check the distance between the the query point and probably all the points, and then whichever will be having the less distance, we will try to select them in that the time complexity is obviously very, very high. That is order of n. So how can we decrease this time complexity. So for this things we will be discussing about two variants that is k d tree and ball tree. Now let's understand what does k d tree basically mean. Let's say that this is my data points okay. And here I have my f1 f2 feature. Based on this particular data points I have plotted this. See I have seven comma two five comma, four nine comma, six two comma, three four comma seven eight comma one. So I have plotted with respect to that. So this one is nothing but eight comma one. This is nothing but seven comma two. This is nothing but two comma three. This is nothing but four five comma four and this is nothing but somewhere around four comma seven. And this is nothing but nine comma six. Okay. So all these points are there. If you want, I can also write it down for you so that you will be able to understand it. So this is seven comma one. Sorry. This is four comma seven okay. Four comma seven. This is nothing but five comma four. And this is nothing but seven comma two. And this is nothing but eight comma one. And this is nothing but nine comma six okay. F1 f2 feature. And this is nothing but two comma three. Now our main aim is that we should definitely find out a way that whenever we try to find out the the k nearest neighbor, right, we should not go ahead and check the distance for each and every point. So for this we will try to create a new tree which is called as K-d tree, which is in the form of a binary tree format. Why? I'll tell you now how this binary tree is basically getting constructed will discuss about this. Okay, so binary tree, what do I mean by binary tree. So this point should will be later converted something like this. See I will try to convert this data points into something like this. Now because of this what will happen is that whenever we do any search, it will be limited to this particular tree itself. Okay, either it can go through this particular path and quickly get the elements, or either it can go through this part and quickly get the elements right. So in short we are reducing the number of searches. Okay. Now let's see how we will probably construct this particular node. Now in the first case what you need to do is that you need to find out what is the median of F1, median of F1, and median of F2 right now. In this case, if I probably see what will be the median. So if I probably try to create this in the sorted manner, it will be two four. Uh, it will be two four, it will be

two, four five. So this is with respect to F1 feature. See this okay. 245789. So here you have around three elements. So what you will do you will take the mid of this. It will be nothing but five plus seven divided by two which is nothing but five plus seven is 12, 12 by two which is nothing but 6.5. Now when I say 6.5, you don't have any elements like 6.5 in the training, right? So either you can take seven or either you can take five, right. Or either you can take see five. Element is present over here with respect to the x axis or seven is present. So either you can go ahead with 5 or 7. You don't have anything like 6.5. Yes we can get the median value to 6.5. But one of them I will actually take. Now let's say that I will be going ahead and taking this seven. Now this seven element is with respect to the x axis. First we will be considering x coordinate right x coordinate basically means this F1 feature. Now once we get this median value what I will do is that I will try to draw a line like this, and I will project it into this x axis. So I'm projecting it over here in the seven value okay. So let's consider this is my seven value okay. So this is the first line I have created now because of this line. Now I have two main regions. This is my region one and this is my region two okay. Now similarly I will go ahead and create my second region. Now in the second region I will go ahead and take my F2. Okay. I will again go ahead and calculate my median value. So here what I will do two for uh I will go ahead and write first in the sorted order. So it will be one comma, two comma, three comma, four and one, two, three, four. And it will also be six comma seven. Now here also you can see there are six elements. So I will try to find out the median of this. Two if I try to find out it will be 3.5. But do I have a element like 3.5. Either it can be four in the y axis or it can be three in the y axis. So let me consider that I am going to consider four in the upper ceiling bracket. So this four. What I will do next is that I will try to. I will try to create another line. And now this time, this will get projected in the y axis. So this is my second line that is getting created whenever I project in the y axis. Okay. It will be in the blue line. Whenever I project in the x axis it will be this particular line. Okay, now this is done. Right now I have actually got this with respect to F1 and F2. Perfect. Now as soon as I divide this line now how many regions I have, I have this one region, I have this two region, I have this three region. Right. And this particular point is basically falling over there. Now similarly what I can do I will take this two points now and I will try to find out the mean with respect to the x axis, right or median with respect to the x axis. I have nine and eight over here. So I can probably consider 8 or 9. Right. And if I probably consider eight or let's say if I consider eight what I will do, I will go ahead and create this another straight line which will divide this two regions again. Right. Which will divide this two regions. Right now why I am dividing this two region. It will make completely sense. I'll just let me explain you in just some time. Okay. Now, similarly, uh, with respect to the x axis, I am able to get something like this. Okay. Over here right now, similarly, for this particular region, I can go ahead and do more divisions. Right? I can make divisions like this or like this. Right. So similarly, like once we go with x and y and x and y continuously, we will be able to make this kind of lines okay. By creating this we are actually creating a we are, we are, we are actually creating what we are creating a split. Right. And when we are creating a split, this split will be also considered over here. Now with respect to this particular point, the first point was seven comma two. Right. Then with respect to seven comma two, I got two region one region was this one region was this. So the second region that I made a split was five comma four over here. Right. Which I projected into the y axis. Now from five comma four I have two plots right four comma seven and two comma three. Now this four comma seven will come over here. And two comma three will come over here. Right. Because two over here is less than five, seven is greater than four. Right. So this kind of split you will be able to see over here. Right. And this is nothing but CD three okay. Now similarly after seven comma two I had two elements nine comma six and eight comma one. Right. So nine comma six will come over here. And eight comma one will come over here. Right. So this way we are able to make the splits right. All the

elements that I had I, I was able to make a simple binary tree over here. Right. Binary tree. And this is nothing but this is a k-d tree. Right in the first time I projected in the x axis. In the second time I projected in the y axis. In the third time we projected in the x axis. And then similarly, when we project in the y axis okay, we can do this right. So now let's see okay, I will also rub this because I don't want to show this because I did not give you a clear idea how we projected this. Right. So I'll just go ahead and delete this. And this will be my two comma three. Let's consider this okay. And this is my four comma seven okay. Four comma seven. This is my two comma three. Now see this. The next line that I projected was this one right in the x axis. This is fine okay. Now after projecting this what happened in with respect to this I got eight nine comma six was one region and a two comma 1 in 1 region. So I made this particular line. Now again if I want to do the split with respect to the y axis, I can again take this particular point because there is only one point, the obviously the median will be this. So I can probably go ahead and construct one line like this. Right now. This will be one group. This will be another group, right. Similarly with respect to four comma seven I can go and project this four over here right in this line. So this line will look something like this. Right. And similarly I can go ahead and continue with my x axis coordinate and y axis. So in short I'm making groups now tomorrow if any new query points come. And now if I want to calculate the distance right. How many elements should I look? I know it is belonging to this region. I know it is belonging to this region. I don't have to worry about any other region. Okay, now this point, let's say this point is nothing but five comma seven. Now where is five comma seven? You know that, right? Five comma seven. First of all, I have to go over here and then I have to go over here right five x axis. There are seven y axis. Now I know what will be the nearest point from here. Either it can be four comma seven or it can be two comma three is not possible. Two comma three is very very far. So I know what will be the my nearest point. That will be nothing but four comma seven. So this becomes my first nearest point. Now if I really want to find out the second one then what I do again I will traverse back. If I traverse back, what is the element I am getting five comma four. So this becomes my second nearest point. Then if I traverse back over here and come in this particular axis, then the third nearest point will become something like this. Right? Again, if I traverse back, I go back again, and then I will be able to see that my next nearest point will be seven comma two. Now similarly, I will go ahead and calculate nine comma six. Then finally I will go ahead and calculate like this. So this way I will be able to get all the nearest point. So over here backtracking is also used. To find out the next nearest point. This was the idea behind CD three. Okay, just to make you understand this thing, why we are doing this. Because we really need to decrease the time complexity. Because with time, I'm just going through one route. I'm not. I'm not finding the distance with respect to every elements that we used to do in the before stage, right? When we don't use CD three, or if we just do a brute search. Brute search basically means go and find out the distance between every point and the other point, right. With respect to this particular query point, if I'm finding the distance with respect to every other point, that becomes a brute search. But now here I'm able to decide my path, and then I'm able to find out my first nearest point, then the second nearest point, and the third nearest point. And obviously backtracking will also happen in this. So this was the entire idea behind the K-d tree. Now similarly, second concept or second type of tree that we specifically have, and this is a better one when compared to K-d tree. Why? Because the in this k-d tree, you know you also have to do back backtracking. But in ball tree which is a second variant, you don't have to do that. Now ball tree concept is very simple. Let's say I have some data points so I can see this is my nearest data point. So this is my one two okay two data point over here I have then I have over here three four. Let's consider like this. And then let's say I have one more 567 okay. Eight nine as the name suggests. Ball tree. Now see in ball tree. The concept is very simple. First of all, what you do, whichever is the

nearest point, right? You group them together. So here you can see two and three are together. Three and four are together. Uh, five, six, seven are together. So I will group them. Eight and nine are together. Okay. So you start your element from here. See? 123456789. Okay. So this is your element, right? Nine. Now what you do first of all what you have combined together a group right. So this is your group one right. So I will combine this two and create a group which is called as group one I will combine three and four I will create another group which is called as G two. And then I will combine five six, seven. I will create a next group which is called as G three. So here you can see this right. And similarly I will go ahead and combine eight and nine. And this will be my group four. So this becomes my group one. Group two, group three and group four right. Now the next step is that what I will do I will try to combine the nearest group nine. Nearest group is nothing but g1, g2 let's say. So this is getting combined right. The next nearest group that you can probably see is G3, G4. So I will go ahead and combine this group right over here. You can see this I'm combining this to this two group. So what happens. First of all I combine G1 and G2 right. So this becomes my G5 group I combine this two. This becomes my G6 group right. And finally I combine the entire group that will be basically become my G7 group. Right. So in short, what we have done, we have combined this with respect to clusters. Now whatever new data point that comes, let's say if I'm getting a new data point over here, what this circle is, this circle is G5. You know, which are which will be the nearest neighbor point, right? The group one neighbor point. So you have all these elements group to neighbor points. So here you directly go ahead and calculate the distance between all this point. So this becomes your top four nearest neighbor right. So this way because of the group we are directly able to get the elements instead of searching for each and every element and calculating the distance. And this is the entire power of Baldry. Okay, so I hope you are able to understand this guy's ball tree concept and k-d tree concept. Right at the end of the day, both are actually creating a binary tree. And always remember with respect to binary tree the time complexity to search or to find the distance. So first of all, to search an element over here and to calculate a distance, it reduces because here you are not trying to calculate a distance with respect to each and every point. Right. So this was an example with respect to k-d tree and ball tree. I hope you are able to understand this. Uh, again, just a way to optimize some kind of algorithms with respect to.