Recall from our last video that it was impossible for us to use hierarchical clustering because of the high resolution of our image.

So in this video, we will try to segment the MRI image using the k-means clustering algorithm.

The first step in k-means clustering involves specifying the number of clusters, k.

But how do we select k?

Well, our clusters would ideally assign each point in the image to a tissue class.

Or a particular substance, for instance, grey matter or white matter, and so on.

And these substances are known to the medical community.

So setting the number of clusters depends on exactly what you're trying to extract from the image.

For the sake of our example, let's set the number of clusters here, k, to five.

And since the k-means clustering algorithm starts by randomly assigning points to clusters, we should set the seed, so that we all obtain the same clusters.

So let's type set.seed, and give it a value of 1.

To run the k-means clustering algorithm, or KMC in short, we need to use the k-means function in R.

And the first input is whatever we are trying to cluster.

In this case it is the healthy vector.

The second argument is the number of clusters, and we can specify it using the argument centers, and that would be equal to k.

And then finally, since the k-means is an iterative method that could take very long to converge, we need to set a maximum number of iterations.

And we can do this by typing iter.max, and give it, for instance, the value 1,000.

And now let's run the k-means algorithm.

The k-means algorithm is actually quite fast, even though we have a high resolution image.

Now to see the result of the k-means clustering algorithm, we can output the structure of the KMC variable.

The first, and most important, piece of information that we get, is the cluster vector.

Which assigns each intensity value in the healthy vector to a cluster.

In this case, it will be giving them values 1 through 5, since we have 5 clusters.

Now recall that to output the segmented image, we need to extract this vector.

The way to do this is by using the dollar notation.

For instance, let us define healthyClusters, and then set it equal to KMC$cluster.

And what we're basically doing here is that we are taking the information, extracting the information of the cluster vector, and putting it in the new variable that is called healthyClusters.

Now how can we obtain the mean intensity value within each of our 5 clusters?

In hierarchical clustering, we needed to do some manual work, and use the t-apply function to extract this information.

In this case, we have the answers ready, under the vector centers.

In fact, for instance, the mean intensity value of the first cluster is 0.48, and the mean intensity value of the last cluster is 0.18.

We can also extract this information using the dollar sign.

For instance, KMC$centers[2].

This should give us the mean intensity value of the second cluster, which is 0.1.

And indeed, this is what we obtain.

Before we move on, I would like to point your attention to one last interesting piece of information that we can get here.

And that is the size of the cluster.

For instance, the largest cluster that we have is the third one, which combines 133,000 values in it.

And interestingly, it's the one that has the smallest mean intensity value, which means that it corresponds to the darkest shade in our image.

Actually, if we look at all the mean intensity values, we can see that they are all less than 0.5.

So they're all pretty close to 0.

And this means that our images is pretty dark.

If we look at our image again, it's indeed very dark.

And we have very few points that are actually white.

Now the exciting part.

Let us output the segmented image and see what we get.

Recall that we first need to convert the vector healthy clusters to a matrix.

To do this, we will use the dimension function, that takes as an input the healthy clusters vector.

And now we're going to turn it into a matrix.

So we have to specify using the combined function, the number of rows, and the number of columns that we want.

We should make sure that it corresponds to the same size as the healthy matrix.

And since we've forgot the number of rows and the number columns in the healthy matrix, we can simply use the nrow and ncol function to get them.

So the first input right now would be nrow of healthy matrix.

And then the second input would be the number of columns of the healthy matrix.

And now we are assigning these numbers of rows and columns to our new matrix, healthy clusters.

And now we can visualize our clusters by using the function image, which takes as an input the healthy cluster's matrix.

And let's turn off the axes.

And then let's be creative and use a fancy color scheme.

We're going to invoke for color here, the rainbow palette in R.

And the rainbow palette, or the function rainbow, takes as an input the number of colors that we want.

In this case, the number of colors would correspond to the number of clusters.

So the input would be k.

And now let's output the segmented image.

Going back to the graphics window, we see that k-means algorithm was able to segment the image in 5 different clusters.

More refinement maybe needs to be made to our clustering algorithm to appropriately capture all the anatomical structures.

But this seems like a good starting point.

The question now is, can we use the clusters, or the classes, found by our k-means algorithm on the healthy MRI image to identify tumors in another MRI image of a sick patient?

We will see if this is possible in the next video.