Applied Predictive Modeling

Lab 1, Part II Transcript: Introduction to R - Data Types

Okay, now let's look at lists, which is a another very important data type in R. So a list, as you can see here, xl is a variable, that is a list.

You call it as list and then provide the variable names. So here it has like three different data types within list. If you can see here, so a is the numeric vector from 1 to 10. Beta is a, you can think of it's like a float which takes exponential -3 to 3. That's about seven values, because it's a sequence from -3 to 3 and then logic is a boolean which is either true or false here.

Let's see how xl looks. As you can see xl is a list of three elements, a, beta, logic. the way to access elements within a list is so you can do it two ways.

If you want to access element a, you can do xl$a and that's going to give you the elements of a or if you want to access the second element in the list which is our beta, you could use the, using these two, two square braces, two square brackets. So xl[[2]] here is going to give you the second element which is the exponential from -3 to +3.

And then you can do the same structure on xl, and it's going to tell you, it's a list of three and it has three elements. One of them is integer, another is numeric, another is logical, and then, and then if you look at the class, as we know, this is a list. So the next most important and the most common data type that you guys are going to deal within this class will be the data frame.

The data frame here, I'm initializing a data frame xd, you can clearly see, it's again the syntax is data.frame. And then you have some columns that you're providing, so here x is a again a numeric vector from 1 to 10, y is letters 1 to 10, what this does is, letters go from a to z, so 1 to 10 will get you the first 10 letters in the alphabets, basically. So that's what y will have. And z is.

Here, what I'm doing is I'm taking true false and then the rep stands for the repeat. So repeat these two elements five times. Basically you'll have 10 elements in z.

So here data frame has all, same number of elements for each of the columns.

Here, the columns are x, y, z, all have 10 elements in them. So let's create this and see how it shows up. So here you have xd, 10 observations of three variables and then you can see, first, first element is called integer, second element is called factor. This is something unique to R that I'm going to go over little bit. Whenever you give a character and especially inside a data frame, it defaults to a factor.

And I think they changed this setting with R 4.0 version and beyond, where strings are by default considered characters, basically strings are not considered as factors.

There is an option called strings as factors that will go, or that determines whether these strings should be considered as factors or just character string or a character vector. So z again, as we know true false are the logical or the boolean kind of values and it just repeats true-false, true-false. So if we look at the structure again, we can clearly see it's a data frame, 10 observations of three variables and it has three columns, x, y, z, each of which, each of which is a different data type within R. One is an integer, one is a factor, one is logical.

And then if you look at the class , it's data frame and head is another useful function. Head usually, by default, gives you the first six rows of the data frame. It's usually used to look at, if you, if the data frame is big and you don't want to look at all the columns and all the rows, in fact, all the rows because you'll look at all the columns. You don't want to print all the rows, head will give you the first six rows to look at it.

As I said, if you don't want y to be a factor and you're using a version of R that doesn't do it by default, one way to do it is you could, use strings as factors is equal to false, and then create the same data frame. So let's create another data frame xd1, where we have strings as factors is equal to false. So here we can see in xd, y was read in as factor, now y is read in as a character vector. So that's the difference.

By just, using this argument strings as factors equal to false, you can change the data type that gets associated to these strings or characters. So again, everything else will remain the same, string str will give you the same thing, head will give you the first six rows, and then names is another useful function that is applied on a data frame.

When you do names , it tells you the names of the columns that are there in the data frame. And then next is the dimension of the data frame. So when you do dimension xd1, as we know, it has 10 rows and three columns. And then nrow is another useful function that tells you how many rows are there. ncol is another useful function that tells you how many columns are there.

So the next question is, how do we access different columns and rows within a data frame?

One way to do it is, you can take xd1[1, so here, the first index is talking about the row and the second index is talking about the column. When I do xd1[1,], it means get me everything, all the columns but only the first row. If you see the output, you'll see, here, this is is one row, and this is the basically what you'll see. And then similarly, if I do xd1[,1], get me all rows, but only get me column one. In this case, column one is this integer vector which has values 1 to 10, so that's what you'll see.

If you want to access a specific element of the dataframe xd1, you'll do xd1 here. You are trying to access row two, column three. So if I execute this column, we know column three is our logical vector and the row two of that is a false. So that's what you see here. And let's say you want to subset your dataframe and get columns, a couple of columns, in this case y and z. The way to do it is you can do this xd1, get me all rows but only columns y and z. When you do that, you can see you have 10 rows of y and z.

And then there is another neat function called subset within the dataframe. So you can create a new dataframe xds here and then all I'm doing is get me the subset of this dataframe where x is greater than five. It's going to get you all the rows that where that condition is satisfied. In this case, if I see, you'll see your x goes from six to 10 which is all greater than five. And another way to subset is like, you just, you can have multiple conditions. You don't need to have only a single condition. Here, I'm subsetting the same dataframe but I'm saying x should be greater than five and z should be equal to true. So if I do this, you can see not only x is greater than five, it's subsetting columns where z is only true.

That's why you see seven and nine and then true, true. So and then not only subsetting, you can select the columns after you do the subset. So here, I'm creating another dataframe xds1 where I'm subsetting xd1 where x is greater than five and I'm selecting columns x and y. So if I do xds1, and then we go here and see, you can see there's only columns x and y and x is always greater than five.

Similarly, the selection can be done another way where you can, instead of saying hey, I want columns x and y, you can say ignore column z so the minus Z is remove column Z and give me everything else.

So when I do this, you can see xds2 is same as xds1 because we only have three columns in this dataframe.

This is a nice handy function subset that you could use to subset data, create new dataframes and is very handy when you are doing your homework problems and other things. Okay, that's our introduction to R with respect to the data types.