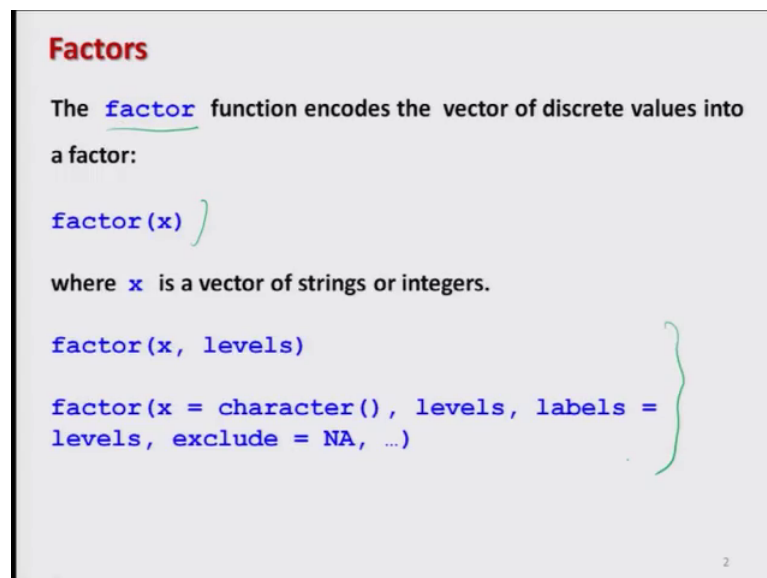


Introduction to R Software
Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture - 23
Factors

Welcome to the next lecture on the course Introduction to R Software. You may kindly recall that in the earlier lecture we started a topic on factors we had understood the concepts of variable, qualitative variable, quantitative variable and then it was extended to the categorical variable and then and from there we had come to the concept of factors in R. We have taken several examples to understand the basic terminologies like as label or say level right be careful with my pronunciation on label and level ok.

(Refer Slide Time: 01:05)



Factors

The `factor` function encodes the vector of discrete values into a factor:

```
factor(x)
```

where `x` is a vector of strings or integers.

```
factor(x, levels)
```

```
factor(x = character(), levels, labels =  
levels, exclude = NA, ...)
```

2

Now, if you try to recall we had use the command `factor` and this was a function which was trying to convert the numerical values in two factors and then we had done some example using `factor` of `x` and we had considered more elaborate.

(Refer Slide Time: 01:26)

```
Factors  
Example  
> x <- factor( c("juice", "juice", "lemonade",  
"juice", "water") )  
>x  
[1] juice  juice  lemonade  juice  water  
Levels: juice lemonade water  
The single levels are ordered alphabetically:  
juice --- lemonade --- water
```

So, this format of factor command and now let us try to take some example and we try to understand more about this factor. So, in the earlier case if you remember we had taken an example, where I had a vector that was taking the values as outcome of the dice and these values were something like 1, 2, 3, 4, 5, 6 some numerical values and we had converted them into a factor. So, number one was getting denoted as one, number two was getting denoted as two and remember one thing both the things were in your control.

So, now in this example I will try to play with the similar type of example and we will try to show you that in case if you are working with the some characters or the string of characters, then how you can do it and this levels or the labels they are in your control and it depends how you want to define them ok.

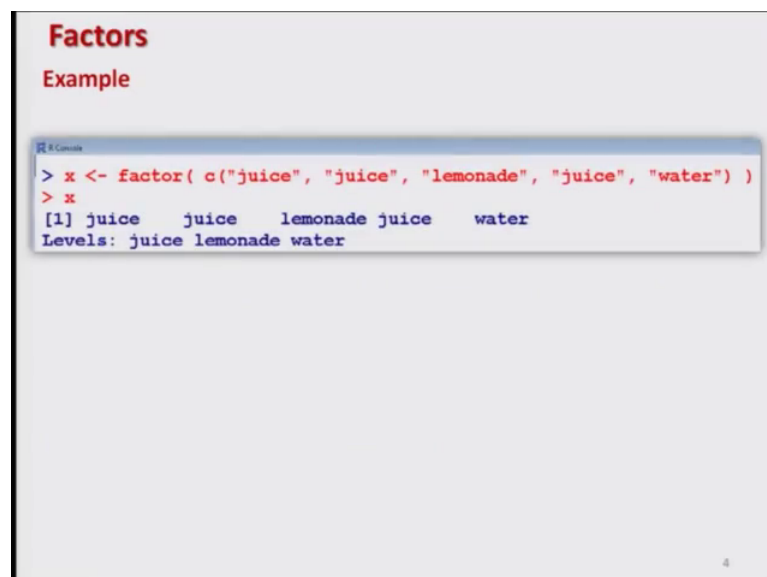
So, now, if you try to see in this example I have simply taken a vector of some strings and this and there are five observations. First observation is say juice second is juice third is lemonade, fourth is juice and fifth here is water and these values are combined by the c operator, and we are interested in finding out the factor of this vector. So, I try to store this outcome in variable here x right and as soon as you enter here on the R console you get here this outcome juice, juice, lemonade, juice water this is the same thing this juice is coming from here, this juice is coming from here, this lemonade is coming from

here, this juice is coming from here and this water is coming from here and it is trying to show you that there are three levels, this is a juice, lemonade and water.

But now in this is statement do you observe anything new? If you try to see here this juice lemonade and water these are my three characters or they are my three strings of characters, which are alphabetically ordered right. If you say here juice comes first as it starts with j then comes lemonade which is start with l, and then comes water that is start with w. So, this is what you have to observe ok.

Now, I try to do the same thing over the R console and let us try to see what happens.

(Refer Slide Time: 04:54)



```
Factors  
Example  
R Console  
> x <- factor( c("juice", "juice", "lemonade", "juice", "water") )  
> x  
[1] juice    juice    lemonade juice    water  
Levels: juice lemonade water
```

You can see here we get the same thing and the screenshot of the same outcome is given over here right.

(Refer Slide Time: 05:20)

Factors

`unclass` function :

All objects in R have a class and function `class` reports it.

For simple vectors, this is just the *mode*, e.g. `"numeric"`, `"logical"`, `"character"`, `"list"`, `"matrix"`, `"array"`, `"factor"` and `"data.frame"`.

A special attribute class of the object is used to allow for an object-oriented style of programming in R.

5

Now, instead of considering some more example, we first need to understand a basic concept that is about a small function which is called as unclass function `unclass` all in a small letters. What is this unclass? Actually we are going to use it further. So, it is important to understand it first. Whenever we are trying to deal with the R software all the data is in the form of some object and every object in R has a class, but how to know what is the class. So, for that we use a function `class` all in a small letters and this class function reports what is the class of an object.

Now, in case if I try to translate it in a simple language, then I will say you all know we already have done one function in more detail recently that is called `mode` mode function, and through mode we is we wanted to have some information what is the mode of the object with there it is numeric, character or list or something like that. So, when we are trying to talk of the simple vectors this class concept is the same concept as of the mode, that is something like it will give us whether the object is numeric, logical, character list matrix or a factor or say data frame. Well we have not done data frame up to now, but we do it in the forthcoming lectures.

So, the next thing is this why do we need this class, why do we need this information on the class. Well at this level we are talking at an very elementary level, but later on when you are writing a bigger program or a or some sensible programming you are doing and suppose you are going for the object oriented programming. Then whenever you are

trying to do the object oriented programming using R, then this information of class is needed. Now depending on the class you have to use an appropriate function.

For example, in case if the data or the object has a different class data frame.

(Refer Slide Time: 08:07)

Factors
`unclass` function

For example if an object has class `"data.frame"`, it will be printed in a certain way, the `plot()` function will display it graphically in a certain way etc.

`unclass()` is used to temporarily remove the effects of class.

Use `help("unclass")` to get more information.

6

Then definitely in case if I want to print a data frame, then we have certain specific commands and in case if this object has a different class say character, we will see later on that in order to print the string of characters we have different types of commands. So, this information on class is very important for us to know in the object oriented programming, because based on that we are going to use the appropriate function to get the required output. So, when we have a concept of class, then in case if I want to do just opposite of that one then also I need some command. So, the opposite of class is unclass that is `unclass` all in a small letters, this function unclass is used to remove the class effects in a temporary way.

For example suppose I have use the class function then obviously, that will have certain effects on my programming. Now I want to temporarily remove the effects of class function. So, I can use the function unclass, and this has to be use with the bracket sign where we give the name of the variable which has to be unclassified; that means, the same variable was class now I want to unclass it. Now in case if you want more information on the on this function unclass, I will say please use this help function to get more

information and we will come back to our original example we which we were considering.

(Refer Slide Time: 10:30)

Factors

`unclass` function

For example if an object has class `"data.frame"`, it will be printed in a certain way, the `plot()` function will display it graphically in a certain way etc.

`unclass()` is used to temporarily remove the effects of class.

Use `help("unclass")` to get more information.

6

(Refer Slide Time: 10:32)

Factors

The command `unclass` shows, an integer is assigned to every factor level:

```
> x <- factor( c("juice" "juice", "lemonade",  
"juice", "water") )
```

strings

```
> unclass(x)  
[1] 1 1 2 1 3  
attr("levels")  
[1] "juice" "lemonade" "water"
```

7

So, now, we have use this data set where I had these value juice, juice lemonade juice and water and we had factorized it and how was the outcome? If you try to see here I can show you in the earlier slide this was here the outcome, and now I want to unclass it. As soon as you type `unclass x` inside the bracket, we get this output `1 1 2 1 3`. And if you try to see what it is trying to do this is simply trying to go back into the reverse direction

from where we started then the concept of factors we had done a one to one mapping between the string of a character and the numerical values. Now, in this case we have a vector of strings of characters these are characters, these are not numerals juice juice lemonade juice and water these are some words these are not the numbers, and now I want to convert this information into a number.

So, if you try to see what are has done in this output 1 1 2 1 3, what is happened that this juice has been classified as number one this is the code given to the word juice number one and. So, thus first value here is one, if you try to see here 1 1 2 1 3, this is my output and the first one is going over here now the next observation is again here juice. So, the code is going to be 1, this is the same code here then the third observation here is lemonade and for lemonade it has given the value 2 and similarly forth value here is juice and this comes over here 1 and similarly fifth value here is water which is denoted by here like this here 3.

So, if you try to see R has automatically given or assigned or number to each of this words juice lemonade and water, and as we had discussed earlier that these levels are arranged in alphabetical order. So, juice is coming at the first position lemonade is coming at the second position and water is coming at the third position because juice starts with j, lemonade you starts with l, and water the start with w. So, they are in the alphabetical order and R has automatically given them the numbers in the same order starting from one. So, juice has been given the code 1, lemonade has been given the code 2, and water has been given the code 3. So, that is the use of say the function unclass let us try to do it over the R software.

(Refer Slide Time: 14:36)

```
> x <- factor( c("juice", "juice", "lemonade", "juice", "water") )
> x
[1] juice    juice    lemonade juice    water
Levels: juice lemonade water
>
> class(x)
[1] "factor"
>
> z = unclass(x)
> z
[1] 1 1 2 1 3
attr("levels")
[1] "juice"    "lemonade" "water"
>
> class(z)
[1] "integer"
> |
```

So, I try to give here the same output. So, you can see here this is my here x. So, now, what you can observe here that this value here x has been stored as a factor, but by looking at this value for example, if I remove each and everything here and if I give you only here x and if I ask you what is the nature of x or in the language of R what is the class of x whether it is numerical value or factor value or say something else we do not know in this case we know because we had given this value x as a factor and you can see this is the outcome here.

Now, if I want to know the class I can simply write down here class of x and we can see here this comes out to be factor and this is what we meant by class right. Now suppose if I want to unclass it and suppose whatever is the outcome, we try to store it say in here z vector unclass here x and outcome here comes out to be like this. So, you can see here this juice has been given the value one, this juice has been given the value one and this lemonade has been given the value here 2 and so on and this is the say this levels.

Now another interesting question comes this x has a class factor, but now what is the class of this here z and remember z has been obtained by using the function unclass over x. you can see here the class of now z is integer. So, what is the model of the story? You had converted a factor which was x into an integer, which is now here an integer. So, now, let us come back to our own slides and try to continue further and this is the screenshot of the outcome that you have just observed.

(Refer Slide Time: 17:20)

```
Factors  
R Console  
> unclass(x)  
[1] 1 1 2 1 3  
attr(,"levels")  
[1] "juice" "lemonade" "water"
```

(Refer Slide Time: 17:28)

```
Factors  
Example  
> x <- factor( c("juice", "juice", "lemonade",  
"juice", "water") )  
>x  
[1] juice juice lemonade juice water  
Levels: juice lemonade water  
The single levels are ordered alphabetically:  
juice --- lemonade --- water
```

Now, I try to make this example a little bit more elaborative, in this example what we had done that we had taken the data as the strings of characters like a juice juice lemonade juice and water and we had operated the function factor on this data set.

And the levels of this factor they were automatically adjusted by or say automatically chosen by the R software for example, you can see here in this outcome, here you can see here that what are the levels that we had obtained here and these levels were obtained as juice lemonade and water. These choice was made automatically by the R software

and suppose I want to now change it I do not want this thing I do not need this thing, but I want to have some other levels depending on my need.

(Refer Slide Time: 18:53)

```
Factors  
If a different assignment is desired, the parameter levels can be used:  
  
> x <- factor( c("juice", "juice", "lemonade",  
"juice", "water"),  
levels=c("water", "juice", "lemonade") )  
  
> x  
[1] juice juice lemonade juice water  
Levels: water juice lemonade  
  
Earlier juice lemonade & water
```

What I do know here that now, I define my own levels and I say my levels are going to be water juice and lemonade. Now you can notice this is not alphabetically ordered earlier it was alphabetically order and then we had juice lemonade and water, but now I want to give it a different level according to my wish or my requirement.

So, I have to simply is specify here levels equal to whatever is my wish for levels which I want to give to this data set, and you can see here as soon as I give hit on the R console we get the same outcome. This juice juice lemonade juice and water this is given here juice juice lemonade juice and water, but this level is now here changed these levels are now as per our requirement and just for your information earlier we had the levels, say juice lemonade and water this was the ordering, but now this ordering is changed. So, if you want to change the order of the levels it is possible.

And in that same example if you want to unclass it, then you get here the this type of vector and it is showing that the levels are now water juice and lemonade; that means, water is indicating here one, juice is indicating here two and lemonade is indicated by three right and if you want to know that what are the levels in this data, you can use the command levels of x and it is giving you the same outcome which is given over here.

So, let us try to do this thing over the R console and can see what happens. So, if I try to give this command over here on the R console, you can see here this is my distinct and if the output here is like this and if I want to unclass it outcome is like this and if I want to find out the levels of x, you can see here this is the thing and the same outcome is given here as a screenshot.

(Refer Slide Time: 21:43)

```
Factors  
Example for an ordered factor:  
> income <- ordered(c("high", "high", "low",  
"medium", "medium"), levels=c("low", "medium",  
"high") )  
  
> income  
[1] high high low medium medium  
Levels: low < medium < high  
  
> unclass(income)  
[1] 3 3 1 2 2  
attr("levels")  
[1] "low" "medium" "high"
```

Now I take another example, suppose I want to have a factor of for a data set in an ordered way or we want to have an ordered factor what is the difference between ordered and say unordered. Suppose I roll a dice in the first throw I get 2, in the second throw I get say here 1 and in the third throw I get 3. So, the order is two one three which is trying to take here the order in which the values have been drawn, but if I say no I am not bothered about the ordering then I will say simply I am observing three values 1 2 and 3 or say 3 1 and 2 or 2 1 and 3 it does not make any difference, but I am more interested only in the three values.

So, now, in case if you want to have some ordering also, then this ordering has to be defined by using a function here `ordered` or `ordered` all in small letters and inside the bracket similar to the use a factor we have to give here the data for example, here I am trying to give the data in terms of high; high low medium and medium and what is the ordering that is given in the see here levels, see here low medium and high and suppose

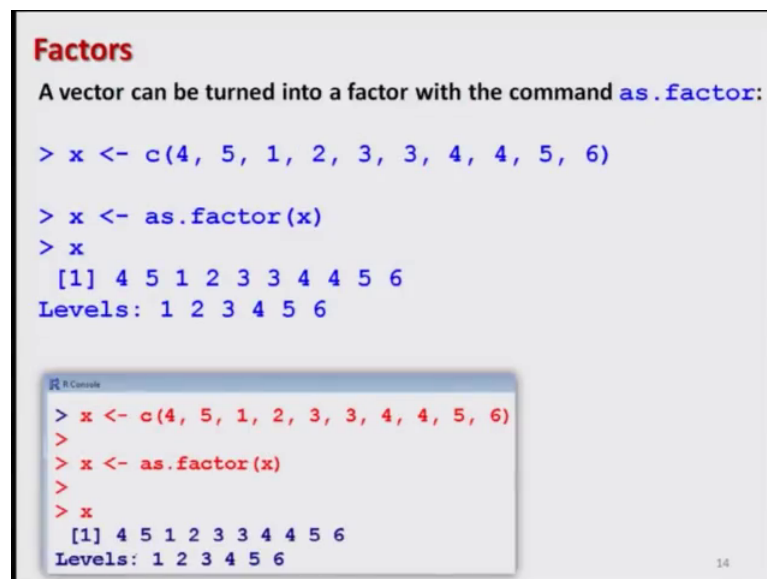
this data on save a high income, low income and medium income is stored in say variable here income.

So, you can see here we get here the same data set whatever is given over here as here, but now this levels whatever I had given here they are coming out in this format that. Low is smaller than medium and medium is smaller than high, and if I want to unclass this variable income we get here this type of thing.

So, let us try to do it over the R console and see what do we obtain here. This is my here income and suppose I want to find out the class of income, this is here ordered factor it is also giving as an the class there it is ordered as well this is factor and suppose I try to unclass my income variable, this is giving us like this and then in case if you want to see what is the class of this unclass income variable. So, I can write down here class of one class income this is coming out to be integer. So, the original thing was ordered factor, but for the class of the after unclasping it is integer right.

So, we come back to our slides now. So, here is the output in this form of a screenshot.

(Refer Slide Time: 25:14)



The screenshot shows an R console window with the following text:

```
Factors  
A vector can be turned into a factor with the command as.factor:  
  
> x <- c(4, 5, 1, 2, 3, 3, 4, 4, 5, 6)  
  
> x <- as.factor(x)  
> x  
[1] 4 5 1 2 3 3 4 4 5 6  
Levels: 1 2 3 4 5 6
```

Below this is a smaller screenshot of the R console showing the same commands and output:

```
> x <- c(4, 5, 1, 2, 3, 3, 4, 4, 5, 6)  
>  
> x <- as.factor(x)  
>  
> x  
[1] 4 5 1 2 3 3 4 4 5 6  
Levels: 1 2 3 4 5 6
```

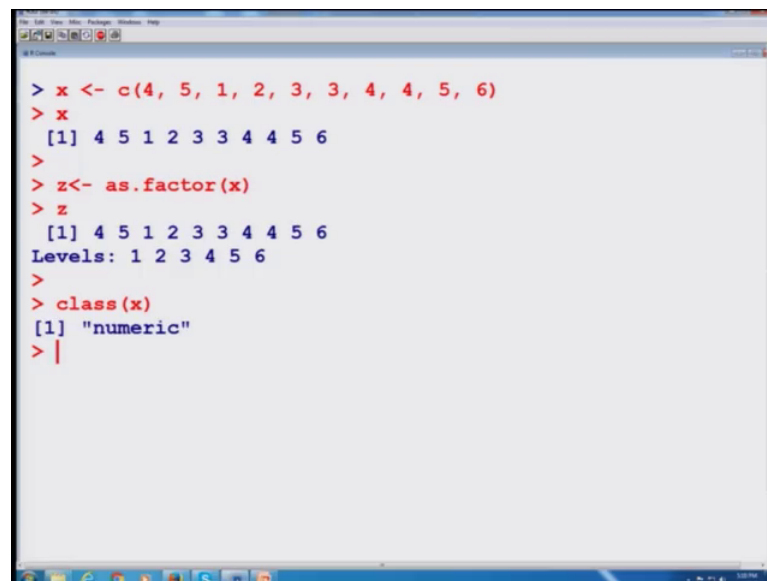
The number 14 is visible in the bottom right corner of the screenshot.

And now if we come to the last topic of this lecture that suppose I want to convert a vector into a factor then we have a command as dot factor. This is a function which can be used to convert a vector into a factor for example, if I try to see this example I am trying to take these numerical values in the form of a vector assigned to x,. And now I

want to convert these numerical values into factor. So, I try to say here as factor x and I try to replace my here x by this here x and now if I try to say here x you can see here this comes out to be a different type of thing that here these are the values, but now it is also giving us the levels.

So, let us try to do it over the R string.

(Refer Slide Time: 26:39)



```
> x <- c(4, 5, 1, 2, 3, 3, 4, 4, 5, 6)
> x
[1] 4 5 1 2 3 3 4 4 5 6
>
> z<- as.factor(x)
> z
[1] 4 5 1 2 3 3 4 4 5 6
Levels: 1 2 3 4 5 6
>
> class(x)
[1] "numeric"
> |
```

So, you can see here this is my here x and now suppose I try to see here the z is my here as factor as dot factor rather, see here x and you can see here the outcome of here is there is this thing. Now in case if you try to see what is the class of x this is numeric. So, this is a numerical vector.

So, now, we stop here and this lecture we have learned some other aspects of the factor. Now, I would request you to look into this lecture in the light of the earlier lecture also try to do some more practice on the outcomes of factor try to experiment with more data sets. And we will see in the next lecture with some new topics, till then goodbye.