390R strings and factors notes
10/20/2020

Today we will talk about the data types.
Ones we already know:
Numeric data types: integer, double
Logical: True and false
String: Contains text in R studio

There are functions you can apply to a string. Ex: find a pattern within strings.
str_sub() function can extract a substring from a value.
str_sub(“Garrett”,1,2)   ---> “Ga”
str_sub(“string”,start, end)   will get the substring starting with start letter and ending with end letter
str_sub(“string”,start) will get a substring starting at start letter.

If you want to change a letter in a string you can assign the string to an object and then assign that object with a new letter at a value. Ex.
g<-”Garrett”
str_sub(g,-1) <- “h” will output “Garreth”

Str_detect will tell you if a string appears within a smaller string
stringa<-c(“string1”,”string2”,”string3”)
str_detect(stringa, “String_to_check_for”) this will check if the string to check for is contained within stringa.

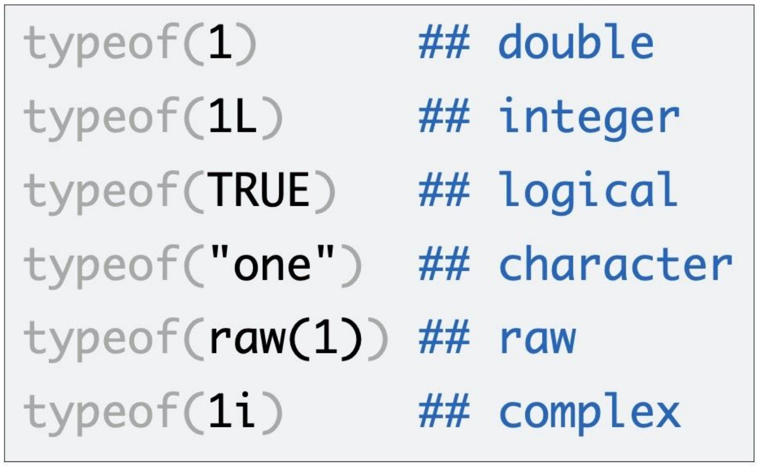
Str_extract will extract a string from a larger string

Question: Do the following steps:
1. Isolate the last letter of every name
2. create a logical variable vowel that displays whether the last letter is one of “a”, “e”, “i”, “o”, or “u”.
3. Use the weighted mean function weighted.mean(vowel, n) to calculate the proportion of children whose name ends in a vowel (by year and sex)
4. and then display the results as a line plot.

First step, extract the last letter of every name.
Then create a variable to check if last letter is a vowel
babynames%>%mutate(last=str_sub(name,-1), vowel = last %in% c(“a”,”i”,”e”,”o”,”u”))
Use weight mean to calculate proportion of children whose name ends in a vowel.
group_by(year,sex)%>%summarize(p_vowel=weighted.mean(vowel,n))
Display results as a line plot
ggplot()+geom_line(aes(x=year,y=p_vowel,color=sex))

When you enter a string in R studio with special characters you might get an error message. If you enter "A/B", you'll get an error because / means something else. If you want to enter "A/B" you need to say "A/n B" or "A//B".
You can use the writeLine function to see what the string displays.
writeLine("A//B")   outputs "A/B"

Atomic types and classes. Only focus on the first 4, last 2 aren't used in this class commonly.



```
typeof(1)        ## double
typeof(1L)       ## integer
typeof(TRUE)     ## logical
typeof("one")    ## character
typeof(raw(1))   ## raw
typeof(1i)       ## complex
```

Factors
Lets create a vector
Eyes <- c(1L, 3L, 3L)   typeof(Eyes) is integer because the L makes the values integers
typeof(c(1,3,3)) is double, no L will be defaulted to a continuous variable
Then create levels for the vector
levels(eyes)<-c("blue","brown","green")   to build a new class, you need to tell the vector the levels to diciate, in this case 3 different colors.
Another way to create a vector is with the factor function
eyes<- factor(x=c("blue","green","green"),levels=c("blue","brown","green"))
You can use factor function to check if values in a level are not contained in a vector.
eyes<- factor(x=c("blue","green","black"),levels=c("blue","brown","green")) would have an NA value for black because it isn't contained in the levels

Using gss_cat dataset
```
gss_cat %>%
  filter(!is.na(tvhours)) %>%                    remove NA values
  group_by(relig) %>%                                  We want to find the tv hours in each religion
  summarize(tvhours = mean(tvhours)) %>%        summarize tv hours average
  ggplot(aes(tvhours, relig)) +                        plot
    geom_point()
```
We can use levels(gss_cat$relig) to get levels

Fct_reorder is used to reorder levels
fct_reorder(factor_to_reorder_by,variable_to_reorder_by, .fun=functino_to_reorder_by, .order)
Ex. ggplot()+geom_point(aes(x=tvhours_avg,y=fct_reorder(relig,tvhours_avg)))
This results in an increasing order in average tv hours for the x-axis.

Question: Make a sensible graph of average TV consumption by marital status.
ggplot()+geom_point(aes(x=tvhours_avg,y=fct_reorder(martial,tvhours_avg)))

Other functions to use
gss_cat%>%ggplot()+geom_bar(aes(fct_infreq(marital))) while reorder the levels in terms of the frequency in decreasing order
gss_cat%>%ggplot()+geom_bar(aes(fct_rev(fct_infreq(marital)))) will reverse the order

Mutate levels, you can rename levels
Ex.
data%>%mutate(partyid,"New var 1 name" = "Old var 1 name", "New var 2 name" = "Old var 2 name")

Collapse multiple levels into 1 levels with fct_collapse
Ex. data%>%mutate(column name=fct_collapse(column name, "new level name"=c("old level1","old level 2"))