STA 445 S24 Assignment 5

Matthew Hashim

3/21/2024

```
library(tidyverse)
```

Problem 1

For the following regular expression, explain in words what it matches on. Then add test strings to demonstrate that it in fact does match on the pattern you claim it does. Do at least 4 tests. Make sure that your test set of strings has several examples that match as well as several that do not. Make sure to remove the `eval=FALSE` from the R-chunk options.

a. This regular expression matches: *This function is looking for any string that contains an a*

```
    strings <- c('a', 'b', 'ab')
    data.frame( string = strings ) %>%
      mutate( result = str_detect(string, 'a') )
```

```
##   string result
## 1      a   TRUE
## 2      b  FALSE
## 3     ab   TRUE
```

b. This regular expression matches: *The one is looking for a string that contains ab*

```
    strings <- c('a', 'b', 'ab', 'abc')
    data.frame( string = strings ) %>%
      mutate( result = str_detect(string, 'ab') )
```

```
##   string result
## 1      a  FALSE
## 2      b  FALSE
## 3     ab   TRUE
## 4    abc   TRUE
```

c. This regular expression matches: *This function is looking for any string that contains either an a, b, or ab*

```
    strings <- c('a', 'b', 'ab', 'abc', 'c')
    data.frame( string = strings ) %>%
      mutate( result = str_detect(string, '[ab]') )
```

```
##   string result
## 1      a   TRUE
## 2      b   TRUE
## 3     ab   TRUE
## 4    abc   TRUE
## 5      c  FALSE
```

d. This regular expression matches: *This one is checking the start of the strings and returning true if the starting char is a or b*

```
strings <- c('a', 'b', 'c', 'ab', 'abc', 'cab')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '^[ab]') )
```

```
##   string result
## 1      a   TRUE
## 2      b   TRUE
## 3      c  FALSE
## 4     ab   TRUE
## 5    abc   TRUE
## 6    cab  FALSE
```

e. This regular expression matches: *This will return true if the string contains a number, a white space, and either an a or A*

```
strings <- c('a', 'b', 'd', 's', 'ab', 'ba', '1 a')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s[aA]') )
```

```
##   string result
## 1      a  FALSE
## 2      b  FALSE
## 3      d  FALSE
## 4      s  FALSE
## 5     ab  FALSE
## 6     ba  FALSE
## 7    1 a   TRUE
```

f. This regular expression matches: *This one is looking for a number, an a or A, and at least 0 white spaces*

```
strings <- c('a', 'a ', '1 a', 'ab', '1a')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s*[aA]') )
```

```
##   string result
## 1      a  FALSE
## 2     a   FALSE
## 3    1 a   TRUE
## 4     ab  FALSE
## 5     1a   TRUE
```

g. This regular expression matches: *This will return true no matter what becuase it is looking for at least 0 of any character*

```
  strings <- c('a', '', 'ab', '1a')
  data.frame( string = strings ) %>%
mutate( result = str_detect(string, '.*') )
```

```
##   string result
## 1      a   TRUE
## 2          TRUE
## 3     ab   TRUE
## 4     1a   TRUE
```

h. This regular expression matches: *This one is looking for exactly a 2 characters that cannot be white spaces followed by exactly bar at the start of the string*

```
  strings <- c('4abar', 'ab', '4a', '4bar', '4aabar', 'aabar', '4 bar', '4abar4')
  data.frame( string = strings ) %>%
    mutate( result = str_detect(string, '^\\w{2}bar') )
```

```
##   string result
## 1  4abar   TRUE
## 2     ab  FALSE
## 3     4a  FALSE
## 4   4bar  FALSE
## 5 4aabar  FALSE
## 6  aabar   TRUE
## 7  4 bar  FALSE
## 8 4abar4   TRUE
```

i. This regular expression matches: *This one is looking for foo.bar any where in the string or the string starts with any 2 char followed by bar*

```
  strings <- c('foo.bar', 'foo.bar1', '1foo.bar', '4abar9999', 'abbar', 'a', 'b')
  data.frame( string = strings ) %>%
    mutate( result = str_detect(string, '(foo\\.bar)|(^\\w{2}bar)') )
```

```
##      string result
## 1   foo.bar   TRUE
## 2  foo.bar1   TRUE
## 3  1foo.bar   TRUE
## 4 4abar9999   TRUE
## 5     abbar   TRUE
## 6         a  FALSE
## 7         b  FALSE
```

Problem 2

The following file names were used in a camera trap study. The S number represents the site, P is the plot within a site, C is the camera number within the plot, the first string of numbers is the YearMonthDay and the second string of numbers is the HourMinuteSecond.

```
    file.names <- c( 'S123.P2.C10_20120621_213422.jpg',
                     'S10.P1.C1_20120622_050148.jpg',
                 'S187.P2.C2_20120702_023501.jpg')
```

```
data <- data.frame(file.names) %>%
  separate(col = file.names, into = c("Site", "Plot", "Camera", "YearMonthDay", "HourMinuteSecond"), sep
  mutate(Year = str_sub(YearMonthDay, 0, 4), Month = str_sub(YearMonthDay, 5, 6),
         Day = str_sub(YearMonthDay, 7, 8), Hour = str_sub(HourMinuteSecond, 0, 2),
         Minute = str_sub(HourMinuteSecond, 3, 4),
         Second = str_sub(HourMinuteSecond, 5, 6),
         HourMinuteSecond = NULL, YearMonthDay = NULL)
data
```

```
##   Site Plot Camera Year Month Day Hour Minute Second
## 1 S123   P2    C10 2012    06  21   21     34     22
## 2  S10   P1     C1 2012    06  22   05     01     48
## 3 S187   P2     C2 2012    07  02   02     35     01
```

Produce a data frame with columns corresponding to the `site`, `plot`, `camera`, `year`, `month`, `day`, `hour`, `minute`, and `second` for these three file names. So we want to produce code that will create the data frame:

```
     Site Plot Camera Year Month Day Hour Minute Second
     S123   P2    C10 2012    06  21   21     34     22
      S10   P1     C1 2012    06  22   05     01     48
      S187   P2     C2 2012    07  02   02     35     01
```

3. The full text from Lincoln's Gettysburg Address is given below. Calculate the mean word length *Note: consider 'battle-field' as one word with 11 letters*).

```
Gettysburg <- 'Four score and seven years ago our fathers brought forth on this
continent, a new nation, conceived in Liberty, and dedicated to the proposition
that all men are created equal. Now we are engaged in a great civil war, testing
whether that nation, or any nation so conceived and so dedicated, can long
endure. We are met on a great battle-field of that war. We have come to dedicate
a portion of that field, as a final resting place for those who here gave their
lives that that nation might live. It is altogether fitting and proper that we
should do this. But, in a larger sense, we can not dedicate -- we can not
consecrate -- we can not hallow -- this ground. The brave men, living and dead,
who struggled here, have consecrated it, far above our poor power to add or
detract. The world will little note, nor long remember what we say here, but it
can never forget what they did here. It is for us the living, rather, to be
dedicated here to the unfinished work which they who fought here have thus far
so nobly advanced. It is rather for us to be here dedicated to the great task
remaining before us -- that from these honored dead we take increased devotion
to that cause for which they gave the last full measure of devotion -- that we
here highly resolve that these dead shall not have died in vain -- that this
nation, under God, shall have a new birth of freedom -- and that government of
the people, by the people, for the people, shall not perish from the earth.'
```

```
mean(str_split(Gettysburg, pattern = "[ ,\\.]")[[1]] %>%
  str_length())
```

```
## [1] 4.047945
```

The mean word length is 4.05 characters