Assignment3

Kyle Bambling

2023-10-26

Question 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. Make sure that your test set of strings has several examples that match as well as several that do not.

a) This regular expression matches: Any string that has the letter a in it

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

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

b) This regular expression matches: Any string with 'ab' in it

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

```
##   string result
## 1      a  FALSE
## 2      b  FALSE
## 3     aa  FALSE
## 4     ab   TRUE
## 5    abc   TRUE
## 6    bac  FALSE
## 7    acb  FALSE
```

c) This regular expression matches: Strings with a or b

```
strings <- c('a', 'b', 'c', 'ab', 'ad', 'bb', 'dc')
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     ad   TRUE
## 6     bb   TRUE
## 7     dc  FALSE
```

d) This regular expression matches: Strings that start with a or b

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

```
##   string result
## 1      a   TRUE
## 2      b   TRUE
## 3      c  FALSE
## 4   dfse  FALSE
## 5  dfsea  FALSE
## 6  adfse   TRUE
```

e) This regular expression matches: A string that has one or more digits, white space, and a or A

```
strings <- c('a', 'b', '345 A', '345 b', '2c', ' a', '2a', '2 a')
data.frame( string = strings ) %>%
mutate( result = str_detect(string, '\\d+\\s[aA]') )
```

```
##   string result
## 1      a  FALSE
## 2      b  FALSE
## 3  345 A   TRUE
## 4  345 b  FALSE
## 5     2c  FALSE
## 6      a  FALSE
## 7     2a  FALSE
## 8    2 a   TRUE
```

f) This regular expression matches: One or more digits, zero or more white space, and a or A

```
strings <- c('a', '345 A', '345 b', '2c', ' a', '2a', '2 a')
data.frame( string = strings ) %>%
mutate( result = str_detect(string, '\\d+\\s*[aA]') )
```

```
##   string result
## 1      a  FALSE
## 2  345 A   TRUE
## 3  345 b  FALSE
## 4     2c  FALSE
## 5      a  FALSE
## 6     2a   TRUE
## 7    2 a   TRUE
```

g) This regular expression matches: Includes zero or more of any character

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

```
##   string result
## 1      a   TRUE
## 2  aaaaa   TRUE
## 3  aaabb   TRUE
```

h) This regular expression matches: A string that starts with 2 alphanumeric characters, then ‘bar’

```
strings <- c('12bar', '11bar', 'barbar', 'bbbar', 'bbar','bbbbar','gfbar')
data.frame( string = strings ) %>%
mutate( result = str_detect(string, '^\\w{2}bar') )
```

```
##   string result
## 1  12bar   TRUE
## 2  11bar   TRUE
## 3 barbar  FALSE
## 4  bbbar   TRUE
## 5   bbar  FALSE
## 6 bbbbar  FALSE
## 7  gfbar   TRUE
```

i) This regular expression matches: A string that is ‘foo.bar’ or starts with 2 alphanumeric characters, then ‘bar’

```
strings <- c('foo.bar', 'foobar', 'ofo.abr', 'wrbar', 'oprab')
data.frame( string = strings ) %>%
mutate( result = str_detect(string, '(foo\\.bar)|(^\\w{2}bar)') )
```

```
##    string result
## 1 foo.bar   TRUE
## 2  foobar  FALSE
## 3 ofo.abr  FALSE
## 4   wrbar   TRUE
## 5   oprab  FALSE
```

Question 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')
```

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:

```
files <- data.frame( file.names = file.names)
files <- files %>% separate(file.names, sep='\\.|_', into=c('site','plot','camera', 'date', 'time'), ren
```

```
## Warning: Expected 5 pieces. Additional pieces discarded in 3 rows [1, 2, 3].
```

```
year = str_sub(files$date, start=1, end=4)
month = str_sub(files$date, start=5, end=6)
day = str_sub(files$date, start=7, end=8)
hour = str_sub(files$time, start=1, end=2)
minute = str_sub(files$time, start=3, end=4)
second = str_sub(files$time, start=5, end=6)
log <- data.frame(
  site = files$site,
  plot = files$plot,
  camera = files$camera,
  year = year,
  month = month,
  day = day,
  hour = hour,
  minute = minute,
  second = second
)
log
```

```
##   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
```

Question 3

The full text from Lincoln's Gettysburg Address is given below. Calculate the mean word length

```
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.'
```

```
sep.strings <- str_replace_all(Gettysburg, pattern=',|\n|--|\\.', replacement = ' ')
sep.strings <- str_replace_all(sep.strings, pattern='-', replacement = '')
sep.strings <- str_split(sep.strings, pattern='\\s+')

words <- data.frame( indiv.word = sep.strings[[1]] )
words <- words %>% mutate(w.length = str_length(indiv.word))
mean(words$w.length)
```

```
## [1] 4.224265
```

Chapter 12

Question 1

Convert the following to date or date/time objects. a) September 13, 2010. b) Sept 13, 2010. c) Sep 13, 2010. d) S 13, 2010. Comment on the month abbreviation needs. e) 07-Dec-1941. f) 1-5-1998. Comment on why you might be wrong. g) 21-5-1998. Comment on why you know you are correct. h) 2020-May-5 10:30 am i) 2020-May-5 10:30 am PDT (ex Seattle) j) 2020-May-5 10:30 am AST (ex Puerto Rico)

```r
#M-D-Y
mdy('September 13, 2010', 'Sept 13, 2010', 'Sep 13, 2010', 'S 13, 2010')
```

```
## [1] "2010-09-13" "2010-09-13" "2010-09-13" "2010-09-13"
```

```r
#D-M-Y
dmy('07-Dec-1941', '1-5-1998', '21-5-1998')
```

```
## [1] "1941-12-07" "1998-05-01" "1998-05-21"
```

```r
#f) It could be January 5th or May 1st
#g) There is no 21st month
ymd_hm('2020-May-5 10:30 am')
```

```
## [1] "2020-05-05 10:30:00 UTC"
```

```r
ymd_hm('2020-May-5 10:30 am', tz= 'US/Pacific')
```

```
## [1] "2020-05-05 10:30:00 PDT"
```

```r
ymd_hm('2020-May-5 10:30 am', tz= 'America/Puerto_Rico')
```

```
## [1] "2020-05-05 10:30:00 AST"
```

Question 2

Using just your date of birth (ex Sep 7, 1998) and today's date calculate the following: a) Calculate the date of your 64th birthday. b) Calculate your current age (in years). c) Using your result in part (b), calculate the date of your next birthday. d) The number of *days* until your next birthday. e) The number of *months* and *days* until your next birthday.

```r
birthday <- ymd('2003-05-08')
#a)
birthday + years(64)
```

```
## [1] "2067-05-08"
```

```r
#b)
age <- year( as.period(birthday %--% ymd('2023-10-24')) )
age
```

```
## [1] 20
```

```r
#c)
next_bday <- birthday + years( age + 1 )
next_bday
```

```
## [1] "2024-05-08"
```

```r
#d)
as.period(ymd('2023-10-24') %--% next_bday, unit = 'days')
```

```
## [1] "197d 0H 0M 0S"
```

```
#e)
as.period(ymd('2023-10-24') %--% next_bday, unit = 'months')
```

```
## [1] "6m 14d 0H 0M 0S"
```

Question 3

Suppose you have arranged for a phone call to be at 3 pm on May 8, 2015 at Arizona time. However, the recipient will be in Auckland, NZ. What time will it be there?

```
call_time <- dmy_hm('08-05-2015 3:00 pm', tz = 'US/Arizona')
with_tz(call_time, tz = 'Pacific/Auckland')
```

```
## [1] "2015-05-09 10:00:00 NZST"
```

Question 5

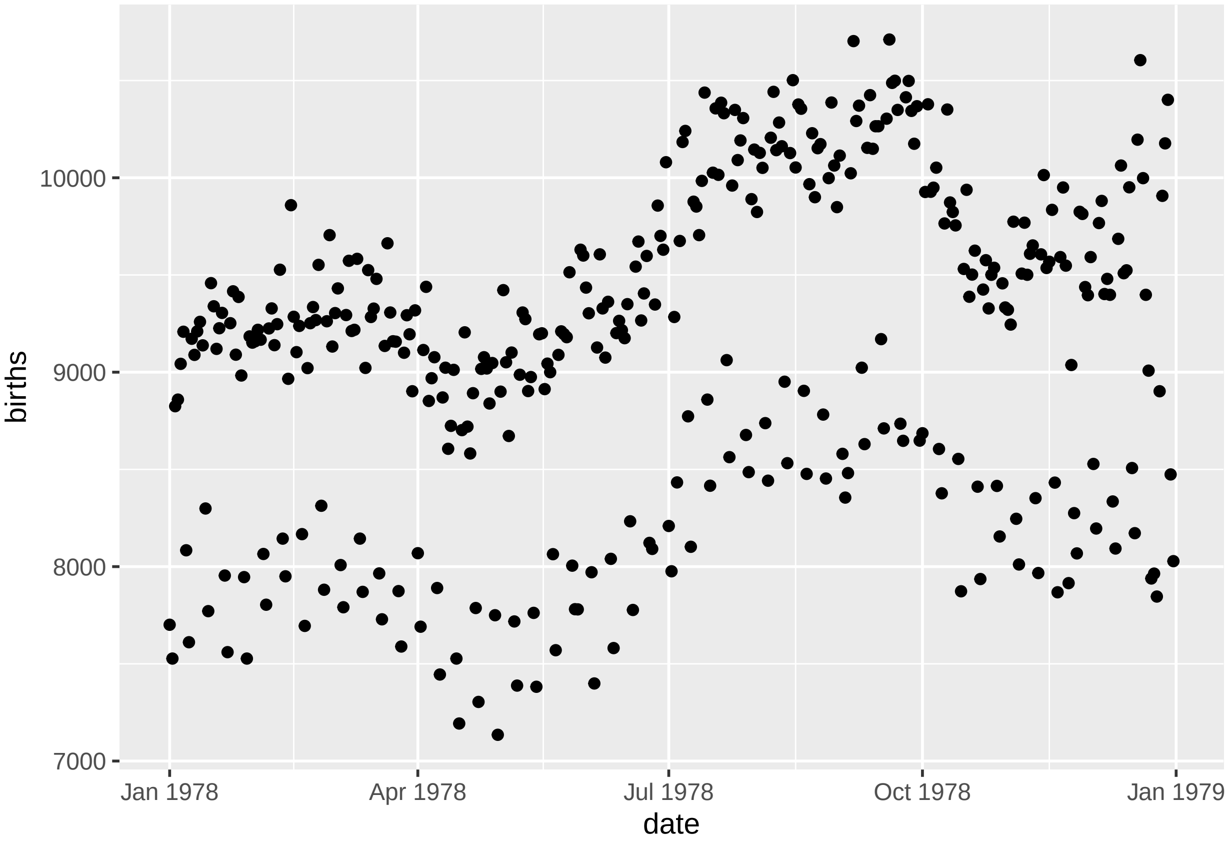
It turns out there is some interesting periodicity regarding the number of births on particular days of the year. a) Using the `mosaicData` package, load the data set `Births78` which records the number of children born on each day in the United States in 1978. Because this problem is intended to show how to calculate the information using the `date`, remove all the columns *except* `date` and `births`.

b) Graph the number of `births` vs the `date` with date on the x-axis. What stands out to you? Why do you think we have this trend?

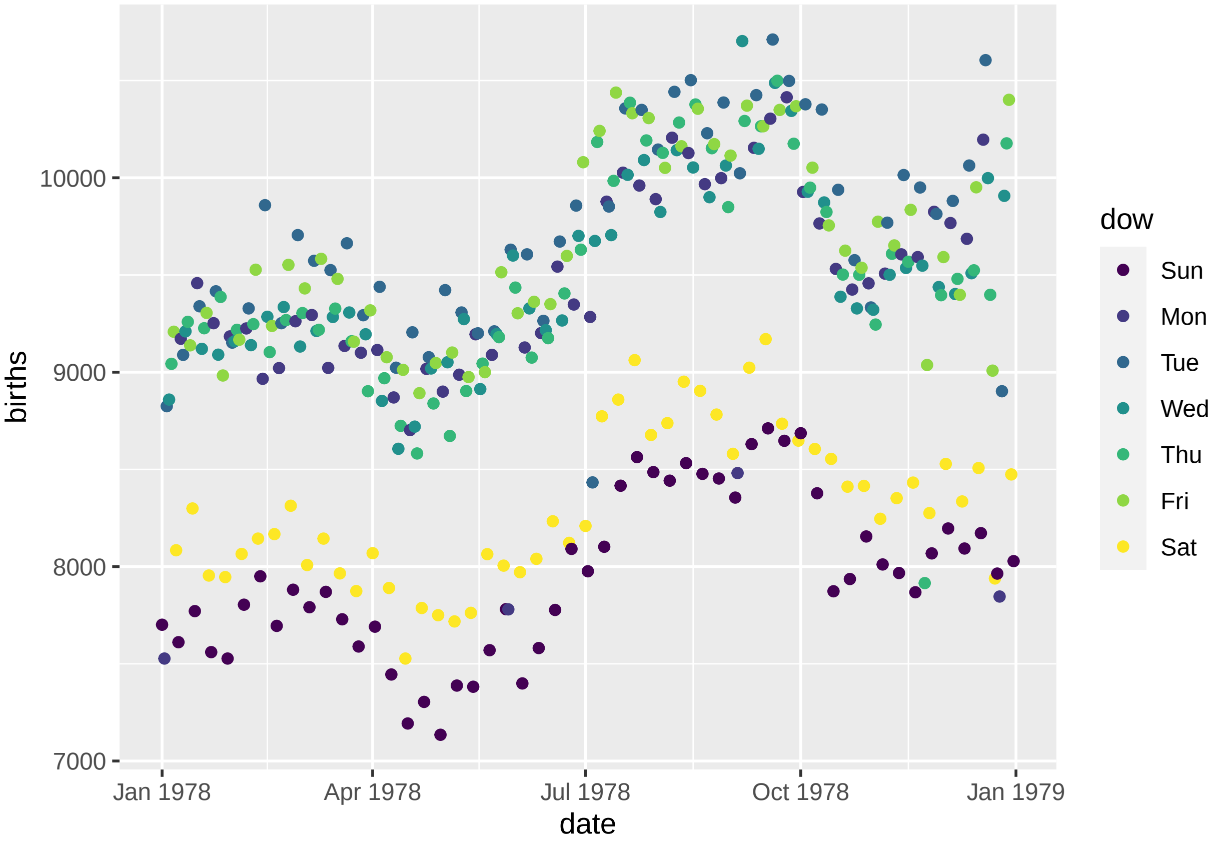
c) To test your assumption, we need to figure out the what day of the week each observation is. Use `dplyr::mutate` to add a new column named `dow` that is the day of the week (Monday, Tuesday, etc). This calculation will involve some function in the `lubridate` package and the `date` column.

d) Plot the data with the point color being determined by the day of the week variable.

```
#a)
birthbydate <- mosaicData::Births78
birthbydate <- birthbydate[-3:-8]
#b)
ggplot(birthbydate, aes(x=date, y=births)) +
  geom_point()
```



```r
#c)
birthbydate <- birthbydate %>% mutate(dow = wday(date, label=TRUE))
#d)
ggplot(birthbydate, aes(x=date, y=births, color=dow)) +
  geom_point()
```


births
10000
9000
8000
7000
Jan 1978
Apr 1978
Jul 1978
Oct 1978
Jan 1979
date
dow
Sun
Mon
Tue
Wed
Thu
Fri
Sat