Assignment Three

Rebecca Chavez

2023-10-17

Chapter Eleven

Question One

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. *If you copy the Rmarkdown code for these exercises directly from my source pages, make sure to remove the* ***`eval=FALSE`*** *from the R-chunk headers.*

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

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

```
##   string result
## 1      a   TRUE
## 2     ab   TRUE
## 3      c  FALSE
## 4  c d a   TRUE
## 5   cdef  FALSE
## 6      A  FALSE
```

b) This regular expression matches: has characters a and b next to each other in order in string

```
# This regular expression matches:  Insert your answer here...
strings <- c('abc', 'cba', 'fabgh', 'adcb', 'a b')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, 'ab') )
```

```
##   string result
## 1    abc   TRUE
## 2    cba  FALSE
## 3  fabgh   TRUE
## 4   adcb  FALSE
## 5    a b  FALSE
```

c) This regular expression matches: has a or b anywhere in string

```
strings <- c('ab', 'ba', 'cab', 'a b', 'ceaceb', 'bet', 'cat', 'cdef')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '[ab]') )
```

```
##   string result
## 1     ab   TRUE
## 2     ba   TRUE
## 3    cab   TRUE
## 4    a b   TRUE
## 5 ceaceb   TRUE
## 6    bet   TRUE
## 7    cat   TRUE
## 8   cdef  FALSE
```

d) This regular expression matches: a or b is at start of string

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

```
##   string result
## 1    abc   TRUE
## 2    bac   TRUE
## 3    cab  FALSE
## 4    acd   TRUE
## 5    bed   TRUE
## 6   cdef  FALSE
## 7    bed  FALSE
```

e) This regular expression matches: at least one digit followed by one white space followed by a or A anywhere in the string

```
strings <- c('1234 A', '23 bac', 'cdefghi 6 abcdef23 i',
             '123\nAa', '84  aA', '93 abcAg', ' aA')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s[aA]') )
```

```
##                 string result
## 1               1234 A   TRUE
## 2               23 bac  FALSE
## 3 cdefghi 6 abcdef23 i   TRUE
## 4              123\nAa   TRUE
## 5               84  aA  FALSE
## 6             93 abcAg   TRUE
## 7                   aA  FALSE
```

f) This regular expression matches: at least one digit followed by any number of white spaces (could be zero) followed by a or A anywhere in the string

```
strings <- c('23a', '23 a', 'abcd23  \n\n abc',
             '   abc', '45 bea')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '\\d+\\s*[aA]') )
```

```
##             string result
## 1              23a   TRUE
## 2             23 a   TRUE
## 3 abcd23  \n\n abc   TRUE
## 4              abc  FALSE
## 5           45 bea  FALSE
```

g) This regular expression matches: any amount of characters or none

```
strings <- c('', 'be', '\n thread', '    ', '\n', '3')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '.*') )
```

```
##      string result
## 1             TRUE
## 2        be   TRUE
## 3 \n thread   TRUE
## 4             TRUE
## 5        \n   TRUE
## 6         3   TRUE
```

h) This regular expression matches: string starts with two alphanumeric characters followed by 'bar', can have anything after this

```
strings <- c('n5bar', '67abr', '75barstool', ' n5bar', 'HEbar 62\np')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '^\\w{2}bar') )
```

```
##        string result
## 1       n5bar   TRUE
## 2       67abr  FALSE
## 3  75barstool   TRUE
## 4       n5bar  FALSE
## 5 HEbar 62\np   TRUE
```

i) This regular expression matches: either 'foo.bar' anywhere in string or string starts with two alphanumeric characters followed by 'bar'

```
strings <- c('foo.bar', '23abfoo.barcab', '67abr', '75barstool',
             ' n5bar', 'HEbar 62\np')
data.frame( string = strings ) %>%
  mutate( result = str_detect(string, '(foo\\.bar)|(^\\w{2}bar)') )
```

```
##           string result
## 1        foo.bar   TRUE
```

```
## 2 23abfoo.barcab   TRUE
## 3          67abr  FALSE
## 4     75barstool   TRUE
## 5          n5bar  FALSE
## 6    HEbar 62\np   TRUE
```

Question Two

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:

‘‘‘r
```
 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
```
‘‘‘

```
# change all delimiters to '.'
file.names <- str_replace_all(file.names, pattern = '_', replacement = '.')

# split file names by delimiter and turn into a matrix
splitData <- str_split_fixed(file.names, pattern = '\\.', n=6)

# assign matrix columns to data frame
cameraData <- data.frame(Site = splitData[,1],
                         Plot = splitData[,2],
                         Camera = splitData[,3],
                         Year = splitData[,4],
                         Month = splitData[,4],
                         Day = splitData[,4],
                         Hour = splitData[,5],
                         Minute = splitData[,5],
                         Second = splitData[,5])

# get correct sub string for date and time values
cameraData <- cameraData %>%
  mutate(Year = str_sub(Year, start = 1, end = 4),
         Month = str_sub(Month, start = 5, end = 6),
         Day = str_sub(Day, start = 7, end = 8),
         Hour = str_sub(Hour, start = 1, end = 2),
         Minute = str_sub(Minute, start = 3, end= 4),
```

```
         Second = str_sub(Second, start = 5, end = 6))

cameraData
```

```
##   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 Three

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.'
```

```
# remove all punctuation
gettysburgCleaned <- str_replace_all(Gettysburg, '\\,|\\.|\\-', '')

# separate by whitespace
subStrings <- str_split(gettysburgCleaned, '\\s+')

# unlist to get lengths
strings <- unlist(subStrings)

# get length of each string and add together to get total
numChars <- sum(str_length(strings))

# get the number of words
numWords <- length(strings)
```

```
# get avg word length
numChars / numWords
```

```
## [1] 4.239852
```

Chapter Twelve

Question One

Convert the following to date or date/time objects.

a) September 13, 2010.

```
mdy("September 13, 2010.")
```

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

b) Sept 13, 2010.

```
# needs a three letter month abbreviation
make_date(year=2010, month=09, day=13)
```

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

c) Sep 13, 2010.

```
mdy('Sep 13, 2010.')
```

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

d) S 13, 2010. Comment on the month abbreviation needs.

```
# one letter month abbreviations will fail since there are multiple
# months with the same first letter
make_date(year=2010, month=09, day=13)
```

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

e) 07-Dec-1941.

```
dmy('07-Dec-1941.')
```

```
## [1] "1941-12-07"
```

f) 1-5-1998. Comment on why you might be wrong.

```r
# could be May 1st or January 5th
mdy('1-5-1998.')
```

```
## [1] "1998-01-05"
```

g) 21-5-1998. Comment on why you know you are correct.

```r
# 21 can only be a day, making it day, month, year order
dmy('21-5-1998.')
```

```
## [1] "1998-05-21"
```

h) 2020-May-5 10:30 am

```r
ymd_hm('2020-May-5 10:30 am')
```

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

i) 2020-May-5 10:30 am PDT (ex Seattle)

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

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

j) 2020-May-5 10:30 am AST (ex Puerto Rico)

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

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

Question Two

Using just your date of birth (ex Sep 7, 1998) and today's date calculate the following *Write your code in a manner that the code will work on any date after you were born.*:

a) Calculate the date of your 64th birthday.

```r
birthday <- mdy('04-02-2002')

# add 64 years to birthday
birthday + years(64)
```

```
## [1] "2066-04-02"
```

b) Calculate your current age (in years). _Hint: Check your age is calculated correctly if your birthda

```
today <- mdy('10-26-2023')

# make time interval from birthday to today
age <- birthday %--% today

# save interval as readable format
age <- as.period(age)
age
```

```
## [1] "21y 6m 24d 0H 0M 0S"
```

d) Using your result in part (b), calculate the date of your next birthday.

```
# get the difference between even 22 years and today's age
# add to today's date to get next birthday
nextBirthday <- today + (years(22) - age)
nextBirthday
```

```
## [1] "2024-04-02"
```

e) The number of _days_ until your next birthday.

```
# time interval from today to next birthday
fromBirthday <- today %--% nextBirthday

# save as readable format in days
as.period(fromBirthday, unit='days')
```

```
## [1] "159d 0H 0M 0S"
```

f) The number of _months_ and _days_ until your next birthday.

```
# save as readable format in months and days
as.period(fromBirthday, unit='months')
```

```
## [1] "5m 7d 0H 0M 0S"
```

Question Three

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?

```
# with_tz() converts same time to new time zone
mdy_hm("May 8, 2015 3:00 pm", tz="US/Arizona") %>%
  with_tz(tz="Pacific/Auckland")
```

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

Question Five

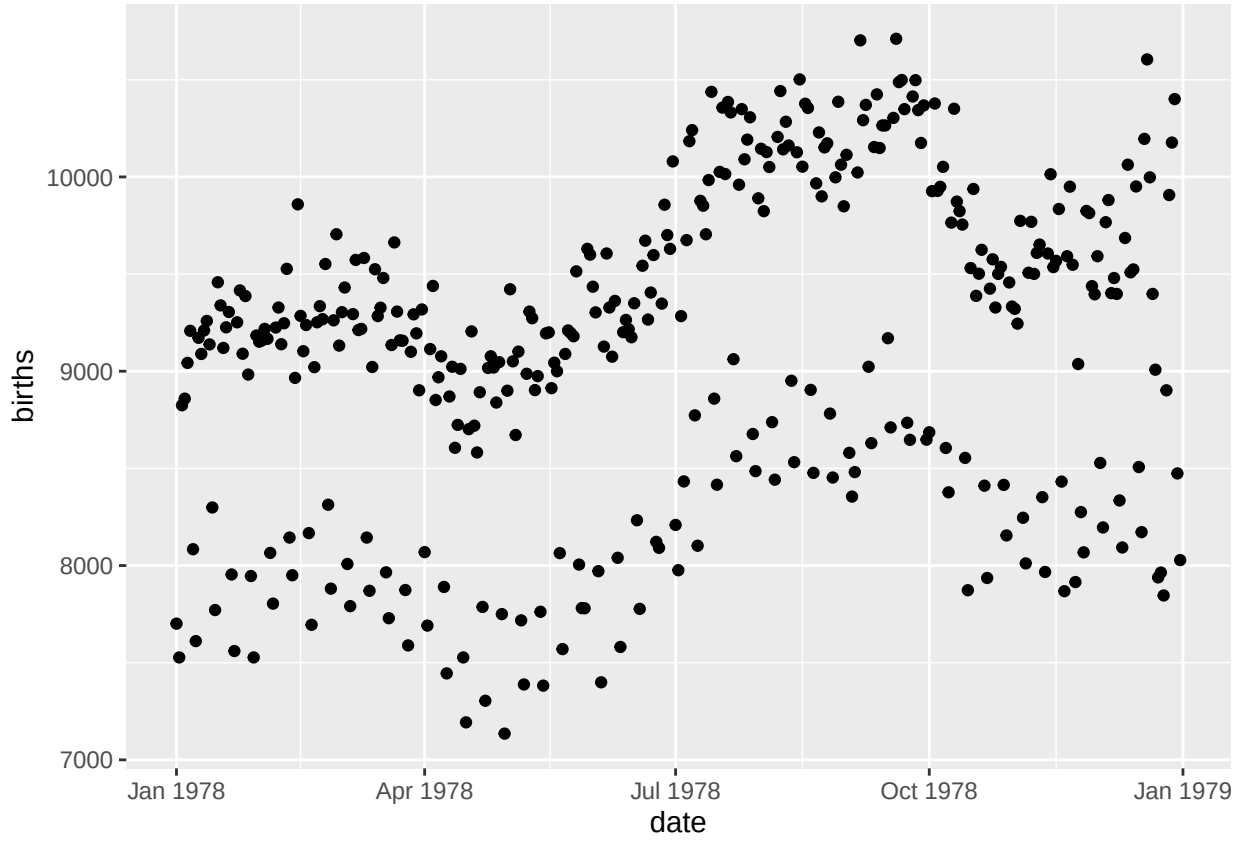
It turns out there is some interesting periodicity regarding the number of births on particular days of the year.

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‘.

```
data('Births78', package='mosaicData')
Births78 <- Births78 %>%
  select(date, 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?

```
ggplot(Births78, aes(y=births, x=date)) +
  geom_point()
```



The number of births alternateds between higher and lower frequently. There may be certain days of the week that tend to have more births than others.

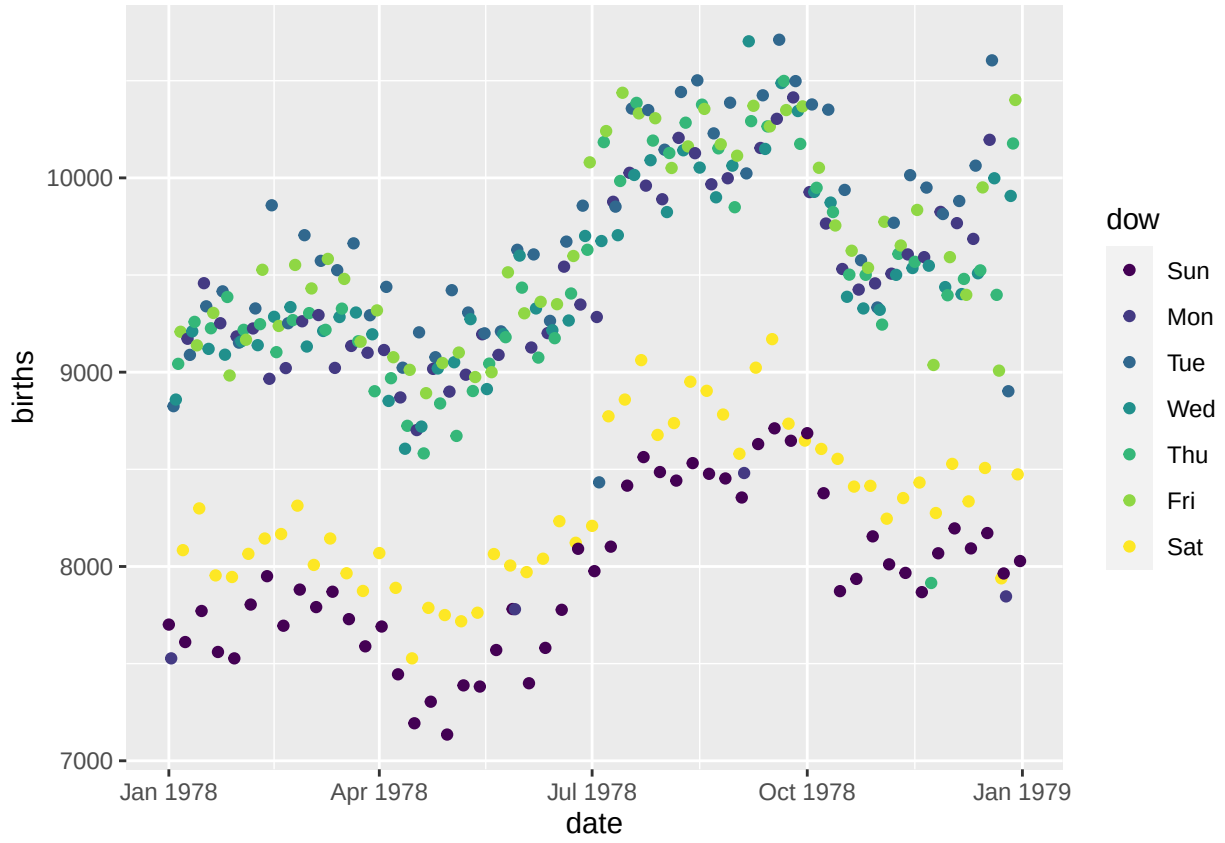
c. To test your assumption, we need to figure out the what day of the week each observation

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.

```
# wday() gives the day of the week as an integer, with 1 being Sunday
# add label = TRUE to get weekday abbreviations
Births78 <- Births78 %>%
  mutate(dow = wday(date, label=TRUE))
```

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

```
ggplot(Births78, aes(y=births, x=date)) +
  geom_point(aes(color=dow))
```



There tend to be less births on Saturday and Sunday than on weekdays.