DA5020 Homework 4: Strings and Factors

2018-02-12

Link to github - https://github.com/GuptaHar/CSR_assignment

Preparation

Download US Farmers Markert Directory data from the website of USDA (click on "Export to Excel"). Rename the file as *farmers_market.csv*.

Download the Know Your Farmer, Know Your Food Projects dataset and name it as *kyfprojects.xls*. Put it into the same folder.

Read the data:

```
library(tidyverse)
library(readxl)



farmers_market<-read_csv("farmers_market.csv")
kyfp<-read_excel("kyfprojects.xls")
```

Warm Up

This dataset stores city and state in different columns, what if you want to print out city and state in the format "City, State"?

```
# Combine city and state separated by ,
farmers_market <- read_csv("farmers_market.csv")
farmers_market$Combined = paste(farmers_market$city, farmers_market$State, se
p=", ")
```

Questions

Please edit this file and add your own solutions to these questions. Make your output as readable as possible. Next time you would need to create this file on your own. Feel free to try out other templates (e.g. Tufte Handout) if your are familiar with LaTex. But for whatever template you choose, you should always include a link to your GitHub repo at the first page of your PDF.

1. (20 points) Cleanup the `Facebook` and `Twitter` column to let them contain only the facebook username or twitter handle name. I.e., replace "https://www.facebook.com/pages/Cameron-Park-Farmers-

Market/97634216535?ref=hl" with "Cameron-Park-Farmers-Market", "https://twitter.com/FarmMarket125th" with "FarmMarket125th", and "@21acres" with "21acres".

```
#Select the data
farmers_market <- read_csv("Farmers_market.csv")
farmers_fb <- select(farmers_market,Facebook)
farmers_to_vector <- as_vector(farmers_fb)
df<-data_frame(facebook = character())

# Clean up the Facebook column using if-else statements
for(i in 1:NROW(farmers_to_vector))
{
  x = farmers_to_vector[[i]]
  a = 'https:[/][/]www.facebook.com[/]pages[/]'
  b = 'https:[/][/]www.facebook.com[/]'
  c = 'www.facebook.com[/]pages[/]'
  d = 'www.facebook.com[/]'
  e = 'facebook.com[/]'
  f = 'http:[/][/]www.'
  g = 'www.'
  h = 'https:[/][/]'
  j = 'http:[/][/]'
  t1 = if (grepl(a,x)==TRUE){gsub(a,"",x)}
  else if(grepl(b,x)==TRUE){gsub(b, "", x)}
  else if(grepl(c,x)==TRUE){gsub(c, "", x)}
  else if(grepl(d,x)==TRUE){gsub(d, "", x)}
  else if(grepl(e,x)==TRUE){gsub(e, "", x)}
  else if(grepl(f,x)==TRUE){gsub(f, "", x)}
  else if(grepl(g,x)==TRUE){gsub(g, "", x)}
  else if(grepl(h,x)==TRUE){gsub(h, "", x)}
  else if(grepl(j,x)==TRUE){gsub(j, "", x)}
  else {x}
  df[i, ] = t1}

# To remove the unwanted numbers from the facebook name at the end
df_test <- gsub("/.*","",as.character(df$facebook))
output_facebook <- data_frame(df_test)

# To Cleanup Twitter
farmers_market_tw <- select(farmers_market,Twitter)
farmers_market_to_vector <- as_vector(farmers_market_tw)
# create dataframe for twitter column
df_T <- data_frame(twitter = character())

for(i in 1:NROW(farmers_market_to_vector))
{
  x= farmers_market_to_vector[[i]]
  p = 'https:[/][/]twitter.com[/]'
  q = 'https:[/][/]www.twitter.com[/]'
  r = '@'
  s = 'www.twitter.com'
  t2 = if (grepl(p,x) == TRUE){gsub(p,"",x)}
  else if(grepl(q,x) == TRUE){gsub(q,"",x)}
  else if(grepl(r,x) == TRUE){gsub(r,"",x)}
  else {x}
  df_T[i, ] = t2}

output_twitter <- df_T

#outputs of the question 1
output_facebook
```

```
## # A tibble: 8,710 x 1
##    df_test
##    <chr>
##  1 Danville.VT.Farmers.Market
##  2 StearnsHomesteadFarmersMarket
##  3 <NA>
##  4 <NA>
##  5 <NA>
##  6 12_South_Farmers_Market
##  7 125thStreetFarmersMarket
##  8 12th-Brandywine-Urban-Farm-Community-Garden
##  9 14UFarmersMarket
## 10 14KennnedyFarmersMarket
## # ... with 8,700 more rows
```

```
output_twitter
```

```
## # A tibble: 8,710 x 1
##    twitter
##  * <chr>
##  1 <NA>
##  2 <NA>
##  3 <NA>
##  4 <NA>
##  5 <NA>
##  6 12southfrmsmkt
##  7 FarmMarket125th
##  8 <NA>
##  9 14UFarmersMkt
## 10 14KenFM
## # ... with 8,700 more rows
```

2. (20 points) Clean up the city and street column. Remove state and county names from the city column and consolidate address spellings to be more consistent (e.g. “St.”, “ST.”, “Street” all become “St”; “and” changes to “&”, etc…).

```
farmers_market_Q2 <- read_csv("farmers_market.csv")
farmers_market_Q2$city <- gsub(" and "," & ",as.character(farmers_market_Q2$c
ity),fixed= TRUE)
# To remove the state name from the city column
farmers_market_Q2$city <- gsub(",.*","",as.character(farmers_market_Q2$city),
fixed= FALSE)
# to conslidate address spelling in the street names
farmers_market_Q2$street<-gsub(" Street"," St",as.character(farmers_market_Q2
$street),fixed = TRUE)
farmers_market_Q2$street<-gsub(" and "," & ",as.character(farmers_market_Q2$s
treet),fixed= TRUE)
farmers_market_Q2$street<-gsub(" street"," St",as.character(farmers_market_Q2
$street),fixed = TRUE)
farmers_market_Q2$street<-gsub(" street "," St",as.character(farmers_market_Q
2$street),fixed = TRUE)
farmers_market_Q2$street<-gsub(" St."," St",as.character(farmers_market_Q2$st
reet),fixed= TRUE)
farmers_market_Q2$street<-gsub(" ST."," St",as.character(farmers_market_Q2$st
reet),fixed = TRUE)
farmers_market_Q2$street<-gsub(" Sts."," St",as.character(farmers_market_Q2$s
treet),fixed= TRUE)
farmers_market_Q2$street<-gsub(" Sts"," St",as.character(farmers_market_Q2$st
reet),fixed = TRUE)
farmers_market_Q2$street<-gsub("ave."," Ave",as.character(farmers_market_Q2$s
treet),fixed= TRUE)
farmers_market_Q2$street<-gsub(" Avenue "," Ave",as.character(farmers_market_
Q2$street),fixed = TRUE)
farmers_market_Q2$street<-gsub(" Avenue"," Ave",as.character(farmers_market_Q
2$street),fixed = TRUE)
farmers_market_Q2$street<-gsub(" Ave"," Ave",as.character(farmers_market_Q2$s
treet),fixed= TRUE)
farmers_market_Q2$street<-gsub(" Ave."," Ave",as.character(farmers_market_Q2$
street),fixed= TRUE)
farmers_market_Q2$street<-gsub(" Boulevard"," Blvd",as.character(farmers_mark
et_Q2$street),fixed= TRUE)
#output of the question 2
farmers_market_Q2

## # A tibble: 8,710 x 59
##       FMID MarketName  Website  Facebook Twitter Youtube OtherMedia street
##      <int> <chr>       <chr>    <chr>    <chr>   <chr>   <chr>      <chr>
##  1 1018261 Caledonia … https:/… https:/… <NA>    <NA>    <NA>       <NA>
##  2 1018318 Stearns Ho… http://… Stearns… <NA>    <NA>    <NA>       6975 …
##  3 1009364 106 S. Mai… http://… <NA>     <NA>    <NA>    <NA>       106 S…
##  4 1010691 10th Steet… <NA>     <NA>     <NA>    <NA>    http://ag… 10th …
##  5 1002454 112st Madi… <NA>     <NA>     <NA>    <NA>    <NA>       112th…
##  6 1011100 12 South F… http://… 12_Sout… @12sou… <NA>    @12southf… 3000 …
##  7 1009845 125th Stre… http://… https:/… https:… <NA>    Instagram… 163 W…
##  8 1005586 12th & Bra… <NA>     https:/… <NA>    <NA>    https://w… 12th …
##  9 1008071 14&U Farme… <NA>     https:/… https:… <NA>    <NA>       1400 …
## 10 1012710 14th & Ken… <NA>     https:/… 14KenFM <NA>    instagram… 5500 …
## # ... with 8,700 more rows, and 51 more variables: city <chr>, County
## #   <chr>, State <chr>, zip <chr>, Season1Date <chr>, Season1Time <chr>,
## #   Season2Date <chr>, Season2Time <chr>, Season3Date <chr>, Season3Time
## #   <chr>, Season4Date <chr>, Season4Time <chr>, x <dbl>, y <dbl>,
## #   Location <chr>, Credit <chr>, WIC <chr>, WICcash <chr>, SFMNP <chr>,
## #   SNAP <chr>, Organic <chr>, Bakedgoods <chr>, Cheese <chr>, Crafts
## #   <chr>, Flowers <chr>, Eggs <chr>, Seafood <chr>, Herbs <chr>,
## #   Vegetables <chr>, Honey <chr>, Jams <chr>, Maple <chr>, Meat <chr>,
## #   Nursery <chr>, Nuts <chr>, Plants <chr>, Poultry <chr>, Prepared
## #   <chr>, Soap <chr>, Trees <chr>, Wine <chr>, Coffee <chr>, Beans <chr>,
## #   Fruits <chr>, Grains <chr>, Juices <chr>, Mushrooms <chr>, PetFood
## #   <chr>, Tofu <chr>, WildHarvested <chr>, updateTime <chr>
```

3. (20 points) Create a new data frame (tibble) that explains the online presence of each state's farmers market. I.e., how many percentages of them have a facebook account? A twitter account? Or either of the accounts? (Hint: use the `is.na()` function)

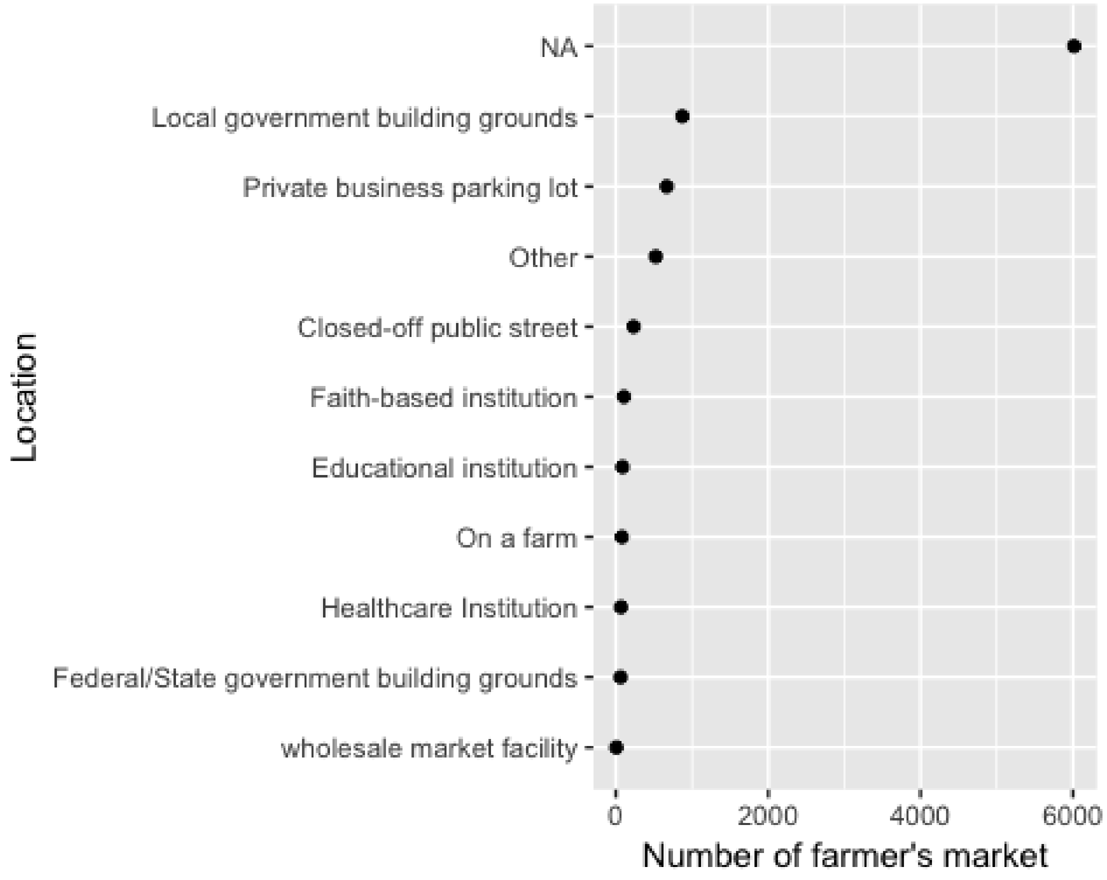
```
library("tibble")
# create a table with online presense for farmers market.
# if a market has facebook profile or have any other online presense it will
be summarised in the table
tibble_table <- read_csv("farmers_market.csv") %>%
  group_by(State) %>%
  summarise( Facebook = ((sum(!is.na(Facebook)))/ n())*100,
             twitter = ((sum(!is.na(Twitter)))/ n())*100,
             Youtube = ((sum(!is.na(Youtube)))/n())*100,
             Other_media = ((sum(!is.na(OtherMedia)))/n())*100,
             Website = ((sum(!is.na(Website)))/n())*100)
tibble_table <- as_data_frame(tibble_table)
class(tibble_table)

## [1] "tbl_df"     "tbl"        "data.frame"

#output of the question 3
output_3 <- tibble_table
output_3

## # A tibble: 53 x 6
##    State                Facebook twitter Youtube Other_media Website
##    <chr>                   <dbl>   <dbl>   <dbl>       <dbl>   <dbl>
##  1 Alabama                  26.1    6.34   0.704        4.93    28.2
##  2 Alaska                   42.1   10.5    0            0       55.3
##  3 Arizona                  56.7   26.7    3.33        15.6     74.4
##  4 Arkansas                 50.9    4.63   1.85         5.56    31.5
##  5 California               40.2   14.2    1.58        12.5     65.0
##  6 Colorado                 42.8    9.43   2.52         2.52    74.8
##  7 Connecticut              31.6   10.3    1.29         7.74    44.5
##  8 Delaware                 62.2   10.8    2.70        18.9     70.3
##  9 District of Columbia     51.7   43.1   22.4         32.8     81.0
## 10 Florida                  42.9    8.43   1.92         4.60    73.6
## # ... with 43 more rows
```

4. (20 points)

Make the location types shorter using the forcats::fct_recode function. Create a plot that demonstrates the number of farmers markets per location type. The locations should be ordered in descending order where the top of the graph will have the one with the highest number of markets

```
farmers_market_Q4 <- read_csv("farmers_market.csv")

# Check unique values of Location types
Unique_location <- distinct(select(farmers_market_Q4,Location))
Unique_location_vector <- as_vector(Unique_location)
Unique_location_vector

##                                                           Location1
##                                                                  NA
##                                                           Location2
##                                      "Private business parking lot"
##                                                           Location3
##                         "Federal/State government building grounds"
##                                                           Location4
##       "On a farm from: a barn, a greenhouse, a tent, a stand, etc"
##                                                           Location5
##                                                             "Other"
##                                                           Location6
## "Faith-based institution (e.g., church, mosque, synagogue, temple)"
##                                                           Location7
##                                          "Closed-off public street"
##                                                           Location8
##                                 "Local government building grounds"
##                                                           Location9
##                         "Co-located with wholesale market facility"
##                                                          Location10
##                                           "Educational institution"
##                                                          Location11
##                                            "Healthcare Institution"

# Using forcats recode function to shorten the location
location_vector_2 <- as_vector(select(farmers_market_Q4, Location))

akl <- recode_factor(location_vector_2, `Faith-based institution (e.g., churc
h, mosque, synagogue, temple)` = "Faith-based institution", `On a farm from:
a barn, a greenhouse, a tent, a stand, etc` = "On a farm", `Co-located with w
holesale market facility` = "wholesale market facility")
akl <- as_data_frame(akl)
akl

## # A tibble: 8,710 x 1
##    value
##    <fct>
##  1 <NA>
##  2 <NA>
##  3 <NA>
##  4 <NA>
##  5 Private business parking lot
##  6 <NA>
##  7 Federal/State government building grounds
##  8 On a farm
##  9 Other
## 10 <NA>
## # ... with 8,700 more rows
```

```
plot_data <- farmers_market_Q4 %>%
  group_by(Location) %>%
  summarise(count = n()) %>%
  as_tibble() %>%
  mutate(Newname = recode_factor(Location, `Faith-based institution (e.g., ch
urch, mosque, synagogue, temple)` = "Faith-based institution", `On a farm fro
m: a barn, a greenhouse, a tent, a stand, etc` = "On a farm", `Co-located wit
h wholesale market facility` = "wholesale market facility"))
ggplot(data = plot_data, mapping = aes(y = reorder(Newname, count) , x = coun
t) )+
  geom_point() + labs(x = "Number of farmer's market") + labs(y = "Location")
```



5. (20 points) Write code to sanity check the `kyfprojects` data. For example, does `Program Abbreviation` always match `Program Name` for all the rows? (Try thinking of your own rules, too.)

```
kyfp <- read_excel("kyfprojects.xls")

# Sanity check 1 - Check if states name are valid
as_data_frame(unique(kyfp$State))

## # A tibble: 56 x 1
##    value
##    <chr>
##  1 IL
##  2 MN
##  3 NM
##  4 LA
##  5 AZ
##  6 WV
##  7 AL
##  8 DC
##  9 NJ
## 10 ND
## # ... with 46 more rows

# Sanity check 2 - Check if year is between 2009 and 2012
year <- grepl("(20[09]\\d|20[12]\\d)", kyfp$Year)
sum(year)

## [1] 2379

table(year)["TRUE"]

## TRUE
## 2379

# Sanity check 3 - Check recipient type
Unique_recipient <- distinct(select(kyfp,'Recipient Type'))
Unique_recipient

## # A tibble: 8 x 1
##   `Recipient Type`
##   <chr>
## 1 Business
## 2 Nonprofit
## 3 Government
## 4 Academic
## 5 Other
## 6 Producer
## 7 Nonprofit Academic
## 8 Businesses
```

```r
# Sanity check 4 - Check Funding type
Unique_funding <- distinct(select(kyfp,'Funding Type'))
Unique_funding
```

```
## # A tibble: 2 x 1
##   `Funding Type`
##   <chr>
## 1 Grant
## 2 Loan
```

```r
# Sanity check 5 - Check Funding amount

maxcheck <- as.numeric(kyfp$'Funding Amount ($)')
maxcheck2 <- na.omit(maxcheck)
  max(maxcheck2)
```

```
## [1] 1e+07
```

```r
# Sanity check 6 - Check Zipcode
zip <- select(kyfp,Zip)
check <- grepl("^[0-9]+$", zip)
check
```

```
## [1] FALSE
```

```r
# Sanity check 7 - Check State characters is limited to 2
check_num <- nchar(kyfp$State)
sum(check_num/2)
```

```
## [1] 2379
```

```r
table(check_num)["2"]
```

```
##    2
## 2379
```

```r
# Sanity check 8 - To Check if town is char
town <- grepl("[A-z]",kyfp$Town)
town
```

```
##    [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [14] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [27] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [40] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [53] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [66] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [79] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##   [92] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [105] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [118] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [131] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [144] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [157] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [170] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [183] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [196] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [209] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [222] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [235] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [248] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [261] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [274] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [287] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [300] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [313] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [326] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [339] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [352] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [365] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [378] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [391] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [404] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [417] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [430] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [443] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [456] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [469] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [482] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [495] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [508] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [521] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [534] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [547] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [560] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [573] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [586] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [599] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [612] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [625] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [638] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [651] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [664] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [677] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [690] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [703] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [716] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [729] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [742] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [755] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [768] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [781] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [794] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [807] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [820] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [833] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [846] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [859] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [872] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [885] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [898] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [911] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [924] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [937] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [950] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [963] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [976] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
##  [989] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1002] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1015] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1028] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1041] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1054] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1067] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1080] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1093] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1106] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1119] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1132] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1145] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1158] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1171] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1184] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1197] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1210] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1223] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1236] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1249] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1262] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1275] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1288] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1301] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1314] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1327] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1340] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1353] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1366] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1379] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1392] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1405] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1418] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1431] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1444] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1457] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1470] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1483] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1496] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1509] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1522] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1535] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1548] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1561] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1574] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1587] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1600] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1613] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1626] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1639] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1652] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1665] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1678] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1691] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1704] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1717] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1730] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1743] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1756] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1769] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1782] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1795] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1808] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1821] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1834] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1847] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1860] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1873] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1886] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1899] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1912] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1925] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1938] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1951] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1964] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1977] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [1990] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2003] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2016] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2029] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2042] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2055] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2068] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2081] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2094] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2107] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2120] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2133] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2146] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2159] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2172] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2185] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2198] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2211] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2224] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2237] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2250] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2263] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2276] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2289] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2302] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2315] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2328] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2341] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2354] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [2367] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Submission

You need to submit an .Rmd extension file as well as the generated pdf file. Be sure to state all the assumptions and give explanations as comments in the .Rmd file wherever needed to help us assess your submission. Please name the submission file LAST_FirstInitial_1.Rmd for example for John Smith’s 1st assignment, the file should be named Smith_J_1.Rmd.