DA5020 - Homework 5: Dates and Times

2019-10-13

Continue working with Farmers Market data from last week.

This week's assignment is not only about dates and times, but also what you learnt from past weeks: data transformation, strings, and more.

You may also need to go through a review on R control statesments since they will come handy in solving some of the problems.

Questions

```
library(readxl)
library(tidyverse)
library(stringr)
library(lubridate)
```

1. (10 points) Add a new column `Season1Days` that contains the number of days a market is opened per week (for the dates it is open).

```
na.vals <- c("", "NA", "n/a", "N/A", "none")
fmarkets <- read.csv("farmers_market.csv", na = na.vals, stringsAsFactors = F)
#kyfp <- read_xls("kyfprojects.xls", na = na.vals)
```

```
#Let's use a regex function to match all weekdays
#Then we count number of grouped matches
reex<-c("([Mm][Oo][Nn])|([Tt][Uu][Ee])|([Ww][Ee][Dd])|([Tt][Hh][Uu])|([Ff][Rr][Ii])|([Ss][Aa][Tt])|([Ss]
fmarkets<-fmarkets%>%
  mutate(
    Season1Days = Season1Time%>%
      str_count(reex),
Season1Days = ifelse(Season1Days == "", NA, Season1Days))
```

2. (10 points) Add a new column `WeekendOpen` indicating whether a market opens during weekends in `Season1`.

```
reex<-c("(([Ss][Aa][Tt])|([Ss][Uu][Nn]))")# Defining a regext to catch saturdays or sundays

# if a store is open in weekends, it prints YES and otherwise it says NO
fmarkets<-fmarkets %>%
  mutate(
   WeekendOpen= Season1Time%>% str_detect(reex))
fmarkets$WeekendOpen<-as.character(fmarkets$WeekendOpen)
fmarkets$WeekendOpen<-str_replace(fmarkets$WeekendOpen,c("TRUE"),"YES")
fmarkets$WeekendOpen<-str_replace(fmarkets$WeekendOpen,c("FALSE"),"NO")
```

3. (20 points) Find out which markets close before 6PM, and which open only for fewer than 4 hours a day. For simplicity, consider only `Season1Time`. For markets with different open hours across a week, use the average length of open hours for the days they actually open.

```
#reex<-c("([0-9]*)[0-9:]{4}) [aApP][mM]") # Defining a regext to catch the time
#fmarkets <- cbind(fmarkets, CloseBefo6=logical(nrow(fmarkets))) # adding a new column to see whther a

# NOw let's write a for loop to see what stores close before 18. If it is TRUE means that the store clo
#for (i in 1:nrow(fmarkets)) { fmarkets[i, 'CloseBefo6'] = max(as.numeric(as.difftime(unlist(str_extrac

head(fmarkets$CloseBefo6,10)
```

```
## NULL
```

```
#partb. We want to see what store are open less than 4 hours a day.

#reex1<-c("([0-9]*)[0-9:]{4}([ ]*)[aApP][mM]([ ]*)[-]([ ]*)([0-9]*)[0-9:]{4}([ ]*)[aApP][mM]|[0-9]{1}[a
#fmarkets <- cbind(fmarkets, averege.open=numeric(nrow(fmarkets)), stringsAsFactors=F)
#fmarkets <- cbind(fmarkets, is.less.4h=character(nrow(fmarkets)), stringsAsFactors=F)

#fmarkets$Season1Time <- fmarkets$Season1Time %>%
  #str_replace("([0-9]{1})([ap][mM])","\\1:00")
#for (i in 1:nrow(fmarkets)) {#i is acting on rows
  #w<-unlist(strsplit(fmarkets$Season1Time[i],";")) # string splitting since all the time are splitted
 # d <- numeric(length(w))
  #if (is.na(w[1])){
  #fmarkets[i, 'is.less.4h'] = "NO DATA"
  #next}
  #for (j in 1:length(w)) {# Avergaing for all the days that a store is open
    #s<-unlist(str_extract_all(w[j],reex1))
    #d[j]<-diff(as.numeric(as.difftime(unlist(strsplit(s, '-')), format="%I:%M %p")))}
  #fmarkets$averege.open[i]<-mean(d)
  #if (fmarkets$averege.open[i]<4){ fmarkets[i, 'is.less.4h'] <- "YES"}
  #else {fmarkets[i, 'is.less.4h'] <- "NO"}}

head(fmarkets$is.less.4h,20)
```

```
## NULL
```

4. (40 Points) The seasons are not standardized and would make analysis difficult. Create four new columns for four seasons (Spring, Summer, Fall, Winter), indicating whether a market is available in that season. Also, create two additional columns `HalfYear` and `YearRound` to identify those who open across seasons. Define "half year" and "year round" on your own terms, but explain them before you write the code (or as comments in your code). (Hint: you may want to create even more auxiliary columns, `Season1BeginDate` and `Season1EndDate` for example.)

```
fmarket.s <- fmarkets[,c(11,13,15,17,19)]# to make our analysis more clear we define a new data fram th
library(dplyr)

#First, we need to clean the data. In this problem we are going to change all the date formats to m/d/y
#Starting with season1date:
fmarket.s$Season1Date <- fmarket.s$Season1Date%>%
  str_remove_all(", [0-9]{4}") %>% # removing  year in format  ", 2019"
str_replace("([jJ]an[.]*[uary]) ([0-9]+)","01/\\2")%>% # Janunary to 01/ plus the second group of patte
str_replace("([fF]eb[.]*[ruary]*) ([0-9]+)","02/\\2")%>%
str_replace("([mM]ar[.]*[ch]*) ([0-9]+)","03/\\2")%>%
```

```
str_replace("([aA]pr[.]*[il]*) ([0-9]+)","04/\\2")%>%
str_replace("([Mm]ay) ([0-9]+)","05/\\2")%>%
str_replace("([Jj]un[.]*[e]*) ([0-9]+)","06/\\2")%>%
str_replace("([jJ]ul[.]*[y]*) ([0-9]+)","07/\\2")%>%
str_replace("([aA]gu[.]*[st]*) ([0-9]+)","08/\\2")%>%
str_replace("([sS]ep[.]*[tember]*) ([0-9]+)","09/\\2")%>%
str_replace("([oO]ct[.]*[ober]*) ([0-9]+)","10/\\2")%>%
str_replace("([nN]ov[.]*[ember]*) ([0-9]+)","11/\\2")%>%
str_replace("([dD]ec[.]*[ember]) ([0-9]+)","12/\\2")

#Same for season 2
fmarket.s$Season2Date <- fmarket.s$Season2Date%>%
  str_remove_all(", [0-9]{4}") %>% # removing  year in format  ", 2019"
str_replace("([jJ]an[.]*[uary]) ([0-9]+)","01/\\2")%>% # Janunary to 01/ plus the second group of patte
str_replace("([fF]eb[.]*[ruary]*) ([0-9]+)","02/\\2")%>%
str_replace("([mM]ar[.]*[ch]*) ([0-9]+)","03/\\2")%>%
str_replace("([aA]pr[.]*[il]*) ([0-9]+)","04/\\2")%>%
str_replace("([Mm]ay) ([0-9]+)","05/\\2")%>%
str_replace("([Jj]un[.]*[e]*) ([0-9]+)","06/\\2")%>%
str_replace("([jJ]ul[.]*[y]*) ([0-9]+)","07/\\2")%>%
str_replace("([aA]gu[.]*[st]*) ([0-9]+)","08/\\2")%>%
str_replace("([sS]ep[.]*[tember]*) ([0-9]+)","09/\\2")%>%
str_replace("([oO]ct[.]*[ober]*) ([0-9]+)","10/\\2")%>%
str_replace("([nN]ov[.]*[ember]*) ([0-9]+)","11/\\2")%>%
str_replace("([dD]ec[.]*[ember]) ([0-9]+)","12/\\2")


#same for season 3
fmarket.s$Season3Date <- fmarket.s$Season3Date%>%
  str_remove_all(", [0-9]{4}") %>% # removing  year in format  ", 2019"
str_replace("([jJ]an[.]*[uary]) ([0-9]+)","01/\\2")%>% # Janunary to 01/ plus the second group of patte
str_replace("([fF]eb[.]*[ruary]*) ([0-9]+)","02/\\2")%>%
str_replace("([mM]ar[.]*[ch]*) ([0-9]+)","03/\\2")%>%
str_replace("([aA]pr[.]*[il]*) ([0-9]+)","04/\\2")%>%
str_replace("([Mm]ay) ([0-9]+)","05/\\2")%>%
str_replace("([Jj]un[.]*[e]*) ([0-9]+)","06/\\2")%>%
str_replace("([jJ]ul[.]*[y]*) ([0-9]+)","07/\\2")%>%
str_replace("([aA]gu[.]*[st]*) ([0-9]+)","08/\\2")%>%
str_replace("([sS]ep[.]*[tember]*) ([0-9]+)","09/\\2")%>%
str_replace("([oO]ct[.]*[ober]*) ([0-9]+)","10/\\2")%>%
str_replace("([nN]ov[.]*[ember]*) ([0-9]+)","11/\\2")%>%
str_replace("([dD]ec[.]*[ember]) ([0-9]+)","12/\\2")

#same for season 4
fmarket.s$Season4Date <- fmarket.s$Season4Date%>%
  str_remove_all(", [0-9]{4}") %>% # removing  year in format  ", 2019"
str_replace("([jJ]an[.]*[uary]) ([0-9]+)","01/\\2")%>% # Janunary to 01/ plus the second group of patte
str_replace("([fF]eb[.]*[ruary]*) ([0-9]+)","02/\\2")%>%
str_replace("([mM]ar[.]*[ch]*) ([0-9]+)","03/\\2")%>%
str_replace("([aA]pr[.]*[il]*) ([0-9]+)","04/\\2")%>%
str_replace("([Mm]ay) ([0-9]+)","05/\\2")%>%
str_replace("([Jj]un[.]*[e]*) ([0-9]+)","06/\\2")%>%
str_replace("([jJ]ul[.]*[y]*) ([0-9]+)","07/\\2")%>%
```

```
str_replace("([aA]gu[.]*[st]*) ([0-9]+)","08/\\2")%>%
str_replace("([sS]ep[.]*[tember]*) ([0-9]+)","09/\\2")%>%
str_replace("([oO]ct[.]*[ober]*) ([0-9]+)","10/\\2")%>%
str_replace("([nN]ov[.]*[ember]*) ([0-9]+)","11/\\2")%>%
str_replace("([dD]ec[.]*[ember]) ([0-9]+)","12/\\2")

#s <- unlist((strsplit(fmarket.s$Season1Date," to ")))
#Now, let's change the format May to October as 05/01 to 10/01 for all 4 seasons
fmarket.s$Season1Date <- fmarket.s$Season1Date%>%
str_replace("[jJ]anuary","01/01")%>%
str_replace("[fF]ebruary","02/01")%>%
str_replace("[mM]arch","03/01")%>%
str_replace("[aA]pril","04/01")%>%
str_replace("[Mm]ay","05/01")%>%
str_replace("[jJ]une","06/01")%>%
str_replace("[jJ]uly","07/01")%>%
str_replace("[aA]gust","08/01")%>%
str_replace("[sS]eptember","09/01")%>%
str_replace("[oO]ctober","10/01")%>%
str_replace("[nN]ovember","11/01")%>%
str_replace("[dD]ecember","12/01")

fmarket.s$Season2Date <- fmarket.s$Season2Date%>%
str_replace("[jJ]anuary","01/01")%>%
str_replace("[fF]ebruary","02/01")%>%
str_replace("[mM]arch","03/01")%>%
str_replace("[aA]pril","04/01")%>%
str_replace("[Mm]ay","05/01")%>%
str_replace("[jJ]une","06/01")%>%
str_replace("[jJ]uly","07/01")%>%
str_replace("[aA]gust","08/01")%>%
str_replace("[sS]eptember","09/01")%>%
str_replace("[oO]ctober","10/01")%>%
str_replace("[nN]ovember","11/01")%>%
str_replace("[dD]ecember","12/01")

fmarket.s$Season3Date <- fmarket.s$Season3Date%>%
str_replace("[jJ]anuary","01/01")%>%
str_replace("[fF]ebruary","02/01")%>%
str_replace("[mM]arch","03/01")%>%
str_replace("[aA]pril","04/01")%>%
str_replace("[Mm]ay","05/01")%>%
str_replace("[jJ]une","06/01")%>%
str_replace("[jJ]uly","07/01")%>%
str_replace("[aA]gust","08/01")%>%
str_replace("[sS]eptember","09/01")%>%
str_replace("[oO]ctober","10/01")%>%
str_replace("[nN]ovember","11/01")%>%
str_replace("[dD]ecember","12/01")

fmarket.s$Season4Date <- fmarket.s$Season4Date%>%
str_replace("[jJ]anuary","01/01")%>%
str_replace("[fF]ebruary","02/01")%>%
```

```
str_replace("[mM]arch","03/01")%>%
str_replace("[aA]pril","04/01")%>%
str_replace("[Mm]ay","05/01")%>%
str_replace("[jJ]une","06/01")%>%
str_replace("[jJ]uly","07/01")%>%
str_replace("[aA]gust","08/01")%>%
str_replace("[sS]eptember","09/01")%>%
str_replace("[oO]ctober","10/01")%>%
str_replace("[nN]ovember","11/01")%>%
str_replace("[dD]ecember","12/01")

#now, the data is cleaned, let's add auxiliary columns as begin date and end dates for different season
fmarket.s <- cbind(fmarket.s, SEASON1BeginDate =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, SEASON1EndDate =character(nrow(fmarkets)), stringsAsFactors=F)

# We split every season column by " to " to establish two additional columns as begin date and end date
for (i in 1:nrow(fmarket.s)) {
  q <- unlist((strsplit(fmarket.s$Season1Date[i]," to ")))
  fmarket.s[i,"SEASON1BeginDate"]=q[1]
  fmarket.s[i,"SEASON1EndDate"]=q[2]
}


#Same trend for all seasons (there are 4 columns)
fmarket.s <- cbind(fmarket.s, SEASON2BeginDate =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, SEASON2EndDate =character(nrow(fmarkets)), stringsAsFactors=F)


for (i in 1:nrow(fmarket.s)) {
  q <- unlist((strsplit(fmarket.s$Season2Date[i]," to ")))
  fmarket.s[i,"SEASON2BeginDate"]=q[1]
  fmarket.s[i,"SEASON2EndDate"]=q[2]
}

fmarket.s <- cbind(fmarket.s, SEASON3BeginDate =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, SEASON3EndDate =character(nrow(fmarkets)), stringsAsFactors=F)


for (i in 1:nrow(fmarket.s)) {
  q <- unlist((strsplit(fmarket.s$Season3Date[i]," to ")))
  fmarket.s[i,"SEASON3BeginDate"]=q[1]
  fmarket.s[i,"SEASON3EndDate"]=q[2]
}

fmarket.s <- cbind(fmarket.s, SEASON4BeginDate =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, SEASON4EndDate =character(nrow(fmarkets)), stringsAsFactors=F)


for (i in 1:nrow(fmarket.s)) {
  q <- unlist((strsplit(fmarket.s$Season4Date[i]," to ")))
  fmarket.s[i,"SEASON4BeginDate"]=q[1]
  fmarket.s[i,"SEASON4EndDate"]=q[2]
}
```

```
# We are removing the year in order to calculat what market is available in what season, later, we defi
fmarket.s$SEASON1BeginDate<-str_replace(fmarket.s$SEASON1BeginDate,"/[0-9]{4}","")
fmarket.s$SEASON1EndDate<-str_replace(fmarket.s$SEASON1EndDate,"/[0-9]{4}","")
fmarket.s$SEASON2BeginDate<-str_replace(fmarket.s$SEASON2BeginDate,"/[0-9]{4}","")
fmarket.s$SEASON2EndDate<-str_replace(fmarket.s$SEASON2EndDate,"/[0-9]{4}","")
fmarket.s$SEASON3BeginDate<-str_replace(fmarket.s$SEASON3BeginDate,"/[0-9]{4}","")
fmarket.s$SEASON3EndDate<-str_replace(fmarket.s$SEASON3EndDate,"/[0-9]{4}","")
fmarket.s$SEASON4BeginDate<-str_replace(fmarket.s$SEASON4BeginDate,"/[0-9]{4}","")
fmarket.s$SEASON4EndDate<-str_replace(fmarket.s$SEASON4EndDate,"/[0-9]{4}","")
#Defining 4 seasons as additinal coulmns
fmarket.s <- cbind(fmarket.s, Spring =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, Summer =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, Fall =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, Winter =character(nrow(fmarkets)), stringsAsFactors=F)


# calculation of intervals for all seasons as we already derived begining and end date of every interva
z1 <- interval(as.Date(fmarket.s$SEASON1BeginDate,"%m/%d"), as.Date(fmarket.s$SEASON1EndDate,"%m/%d"))

z2<- interval(as.Date(fmarket.s$SEASON2BeginDate,"%m/%d"), as.Date(fmarket.s$SEASON2EndDate,"%m/%d"))

z3 <- interval(as.Date(fmarket.s$SEASON3BeginDate,"%m/%d"), as.Date(fmarket.s$SEASON3EndDate,"%m/%d"))

z4 <- interval(as.Date(fmarket.s$SEASON4BeginDate,"%m/%d"), as.Date(fmarket.s$SEASON4EndDate,"%m/%d"))

k1 <- interval(as.Date("03/01","%m/%d"), as.Date("05/30","%m/%d"))# Spring period
k2 <- interval(as.Date("06/01","%m/%d"),as.Date("08/30","%m/%d"))#Summer period
k3 <- interval(as.Date("09/01","%m/%d"),as.Date("11/30","%m/%d"))#Fall

k4 <- interval(as.Date("12/01","%m/%d"), as.Date("12/30","%m/%d"))#Winterpart1

k5 <- interval(as.Date("01/01","%m/%d"), as.Date("02/28","%m/%d"))#Winterpart2

#We calculate for all seasons, like spring all the days that a market is open. we assign zero to NA val
p1<- day(as.period(intersect(k1, z1),"days"))
p1[is.na(p1)] <- 0
p2<- day(as.period(intersect(k1, z2),"days"))
p2[is.na(p2)] <- 0
p3<- day(as.period(intersect(k1, z3),"days"))
p3[is.na(p3)] <- 0
p4<- day(as.period(intersect(k1, z4),"days"))
p4[is.na(p4)] <- 0
fmarket.s$Spring <- p1+p2+p3+p4

q1<- day(as.period(intersect(k2, z1),"days"))
q1[is.na(q1)] <- 0
q2<- day(as.period(intersect(k2, z2),"days"))
q2[is.na(q2)] <- 0
q3<- day(as.period(intersect(k2, z3),"days"))
q3[is.na(q3)] <- 0
q4<- day(as.period(intersect(k2, z4),"days"))
q4[is.na(q4)] <- 0
fmarket.s$Summer <- q1+q2+q3+q4
```

```
r1<- day(as.period(intersect(k3, z1),"days"))
r1[is.na(r1)] <- 0
r2<- day(as.period(intersect(k3, z2),"days"))
r2[is.na(r2)] <- 0
r3<- day(as.period(intersect(k3, z3),"days"))
r3[is.na(r3)] <- 0
r4<- day(as.period(intersect(k3, z4),"days"))
r4[is.na(r4)] <- 0
fmarket.s$Fall <- r1+r2+r3+r4

s1<- day(as.period(intersect(k4, z1),"days"))
s1[is.na(s1)] <- 0
s2<- day(as.period(intersect(k4, z2),"days"))
s2[is.na(s2)] <- 0
s3<- day(as.period(intersect(k4, z3),"days"))
s3[is.na(s3)] <- 0
s4<- day(as.period(intersect(k4, z4),"days"))
s4[is.na(s4)] <- 0

t1<- day(as.period(intersect(k5, z1),"days"))
t1[is.na(t1)] <- 0
t2<- day(as.period(intersect(k5, z2),"days"))
t2[is.na(t2)] <- 0
t3<- day(as.period(intersect(k5, z3),"days"))
t3[is.na(t3)] <- 0
t4<- day(as.period(intersect(k5, z4),"days"))
t4[is.na(t4)] <- 0
fmarket.s$Winter <- s1+s2+s3+s4+t1+t2+t3+t4

# For the halfyear and full year calculations, we define a column as the total number of days that a st
fmarket.s <- cbind(fmarket.s, HalfYear =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, YearAround =character(nrow(fmarkets)), stringsAsFactors=F)
fmarket.s <- cbind(fmarket.s, TotalOpenDays =character(nrow(fmarkets)), stringsAsFactors=F)

fmarket.s$TotalOpenDays <- fmarket.s$Spring+fmarket.s$Summer+fmarket.s$Fall+fmarket.s$Winter

fmarket.s$HalfYear <-  ifelse(fmarket.s$TotalOpenDays<200 & fmarket.s$TotalOpenDays>160, "Yes", "No")
fmarket.s$YearAround <-  ifelse( fmarket.s$TotalOpenDays>340, "Yes", "No")







#"days"))+day(as.period(intersect(k1, z2), "days"))+day(as.period(intersect(k1, z3), "days"))+day(as.pe


#fmarket.s$Spring<-  (k1 %within% z1)

#|(k1 %within% z2)|(k1 %within% z3)|(k1 %within% z4)
```

5. (20 points) *Open question*: explore the new variables you just created. Aggregate them at different geographic levels, or some other categorical variable. What can you discover?

```r
#As an example, in problem 1, we have already calculated season 1 days, now lets average them over the
a <- fmarkets %>% group_by(State) # grouping the data by state
b <- summarize(a,mean.open.day=mean(Season1Days,na.rm=TRUE))

head(b,10)
```

```
## # A tibble: 10 x 2
##    State                mean.open.day
##    <chr>                        <dbl>
##  1 Alabama                       1.62
##  2 Alaska                        1.22
##  3 Arizona                       1.07
##  4 Arkansas                      1.74
##  5 California                    1.07
##  6 Colorado                      1.12
##  7 Connecticut                   1.03
##  8 Delaware                      1.27
##  9 District of Columbia          1.08
## 10 Florida                       1.42
```

```r
#Now let's use tha package ggmap and maps function to visualize the data for 3 states as an example.
library("maps")
map.text("state", regions=c("California","Alabama","Florida"), labels=c("1.072","1.61","1.41"))
```

```r
# Now let's do averaging over the whole year
a1 <- fmarket.s %>% group_by(State) # grouping the data by state
b1 <- summarize(a1,mean.open.year=mean(TotalOpenDays,na.rm=TRUE))

head(b1,10)
```

```
## # A tibble: 10 x 2
##    State                mean.open.year
##    <chr>                         <dbl>
##  1 Alabama                        73.3
##  2 Alaska                         81.2
##  3 Arizona                       175.
##  4 Arkansas                      118.
##  5 California                    151.
##  6 Colorado                       83.9
##  7 Connecticut                    48.6
##  8 Delaware                      103.
##  9 District of Columbia          148.
## 10 Florida                       163.
```

```r
map.text("state", regions=c("California","Alabama","Florida"), labels=c("151","73","163"))
```


```r
Open.Spring <- fmarket.s %>% group_by(State) # grouping the data by state
OSP <- summarize(Open.Spring,mean.open.SPRING=mean(Spring,na.rm=TRUE))
```

```r
Open.Summer <- fmarket.s %>% group_by(State) # grouping the data by state
OSU <- summarize(Open.Summer,mean.open.Summer=mean(Summer,na.rm=TRUE))

Open.Fall <- fmarket.s %>% group_by(State) # grouping the data by state
OF <- summarize(Open.Fall,mean.open.Fall=mean(Fall,na.rm=TRUE))

Open.Winter <- fmarket.s %>% group_by(State) # grouping the data by state
OW <- summarize(Open.Winter,mean.open.Winter=mean(Winter,na.rm=TRUE))

df1<-merge(x=OSP,y=OSU,by="State")
df2<-merge(x=OF,y=OW,by="State")
df<-merge(x=df1,y=df2,by="State")

head(df,10)
```

```
##                   State mean.open.SPRING mean.open.Summer mean.open.Fall
## 1               Alabama        11.671429         37.97857       18.40714
## 2                Alaska         8.864865         53.16216       14.45946
## 3               Arizona        36.817204         74.25806       41.46237
## 4              Arkansas        23.045045         53.12613       32.90090
## 5            California        34.670619         51.61528       39.41765
## 6              Colorado         7.256250         49.22500       23.87500
## 7           Connecticut         3.360759         25.97468       17.55696
## 8              Delaware        11.666667         61.47222       27.16667
## 9  District of Columbia        19.758621         69.34483       53.75862
## 10              Florida        39.897727         51.69318       41.43182
##    mean.open.Winter
## 1          5.221429
## 2          4.702703
## 3         22.795699
## 4          8.711712
## 5         25.498024
## 6          3.550000
## 7          1.664557
## 8          2.416667
## 9          4.948276
## 10        30.393939
```

```r
#If we compare average of opening days for different states, It can be seen that in cold states like Al
```

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.