# homework2

Huang Xubin 3180102999

2021/7/12

## 0

```
library(MASS)
library(lattice)
library(Devore7)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
library(ggplot2)
library(readr)
```

## 1

### a. Load the data into a dataframe called ca_pa.

```
invisible(ca_pa <- read_csv("data\\calif_penn_2011.csv"))
```

```
## Warning: Missing column names filled in: 'X1' [1]

##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   STATEFP = col_character(),
##   COUNTYFP = col_character(),
##   TRACTCE = col_character(),
##   GEO.display.label = col_character()
## )
## i Use `spec()` for the full column specifications.
```

### b. How many rows and columns does the dataframe have?

There are 11,275 rows and 34 columns in this data frame, which is specified in the very first line of the message below.

```
# str(ca_pa)
```

### c. Run this command, and explain, in words, what this does: colSums(apply(ca_pa,c(1,2),is.na))

This command first apply the method "is.na" upon each element in data frame "cp_pa" and then do the column sum, which makes it clear how na values distributed w.r.t. the columns.

```
# colSums(apply(ca_pa,c(1,2),is.na))
```

### d. The function na.omit() takes a dataframe and returns a new dataframe, omitting any row containing an NA value. Use it to purge the data set of rows with incomplete data.

```
ca_pa <- na.omit(ca_pa)
# colSums(apply(ca_pa,c(1,2),is.na))
```

### e. How many rows did this eliminate?

According to the information below, 670 rows have been eliminated.

```
# str(ca_pa)
eliminated_row_num <- 11275 - 10605
eliminated_row_num
```

```
## [1] 670
```

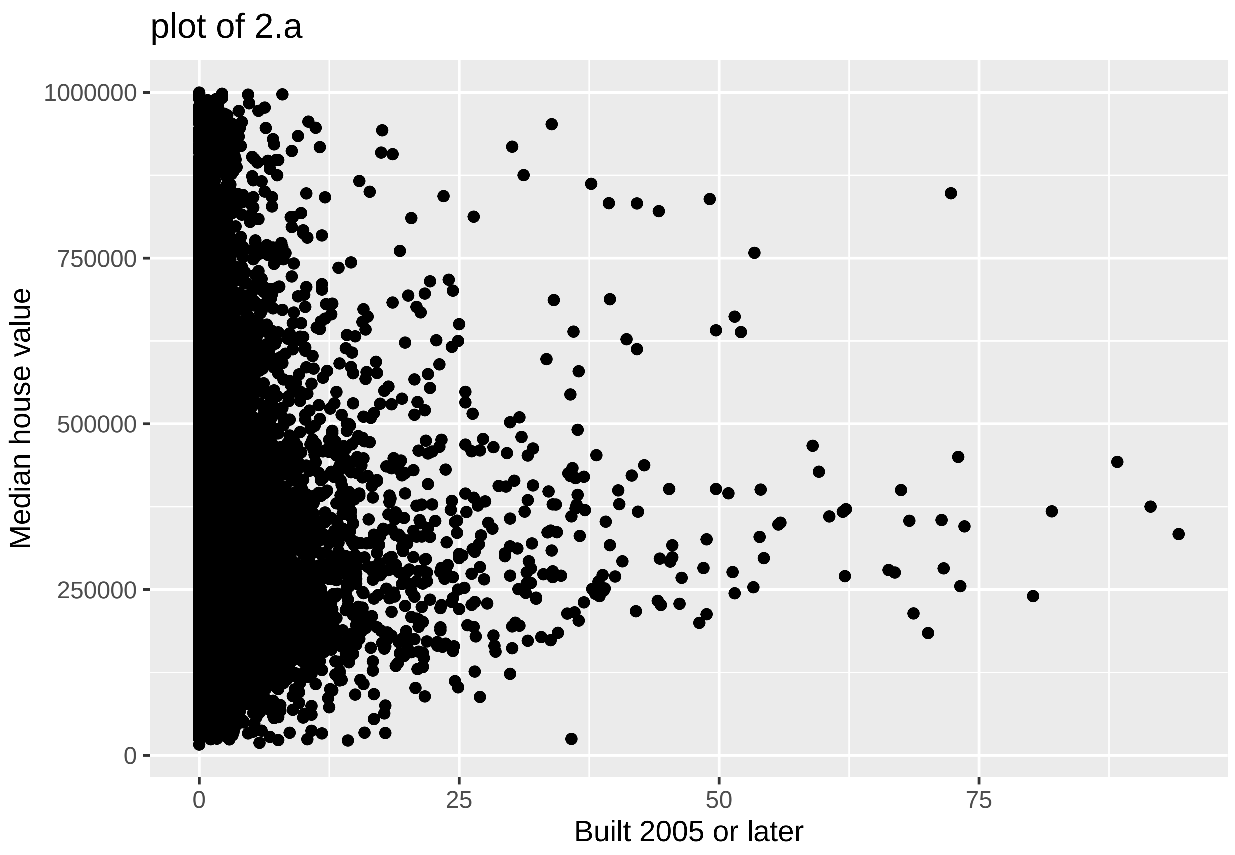
### f. Are your answers in (c) and (e) compatible? Explain.

It is compatible. Since one row can take more than just one NA value among its columns, the number of NA elements of each column does not necessarily stands for the number of rows eliminated.

## 2

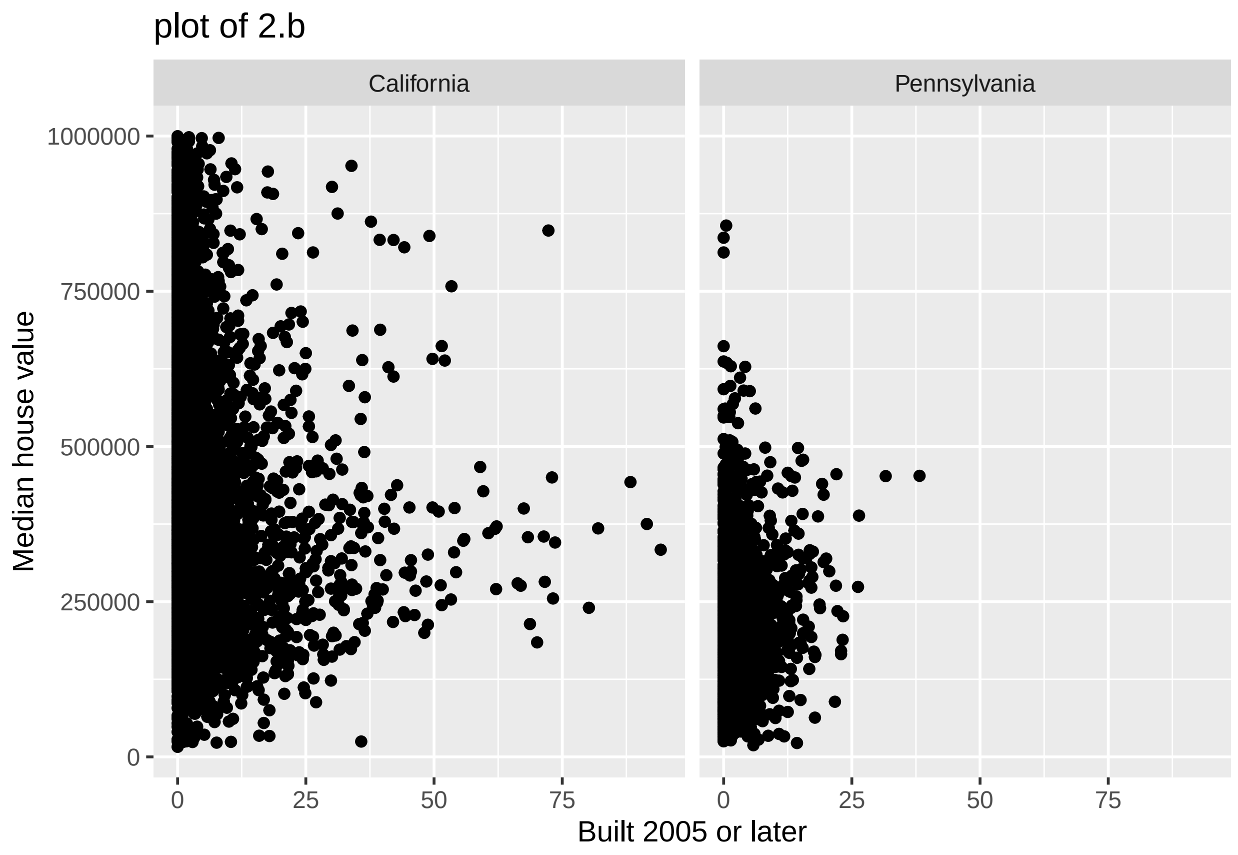
### a. The variable Built_2005_or_later indicates the percentage of houses in each Census tract built since 2005. Plot median house prices against this variable.

```
ca_pa %>% ggplot() +
  geom_point(aes(x = Built_2005_or_later, y = Median_house_value)) +
  labs(x = "Built 2005 or later", y = "Median house value",
       title = "plot of 2.a")
```



**b. Make a new plot, or pair of plots, which breaks this out by state. Note that the state is recorded in the STATEFP variable, with California being state 6 and Pennsylvania state 42.**

```r
names <- c('06' = "California", '42' = "Pennsylvania")
ca_pa %>% ggplot() +
  geom_point(aes(x = Built_2005_or_later, y = Median_house_value)) +
  labs(x = "Built 2005 or later", y = "Median house value",
       title = "plot of 2.b") +
  facet_wrap(~STATEFP, labeller = as_labeller(names))
```



## 3

**a. Add a new column to the dataframe which contains the vacancy rate. What are the minimum, maximum, mean, and median vacancy rates?**

```
invisible(ca_pa <- ca_pa %>% mutate(Vacancy_rate = Vacant_units / Total_units))
str(ca_pa$Vacancy_rate)
```

```
##  num [1:10605] 0.0398 0.0505 0.0356 0.0417 0.0832 ...
```

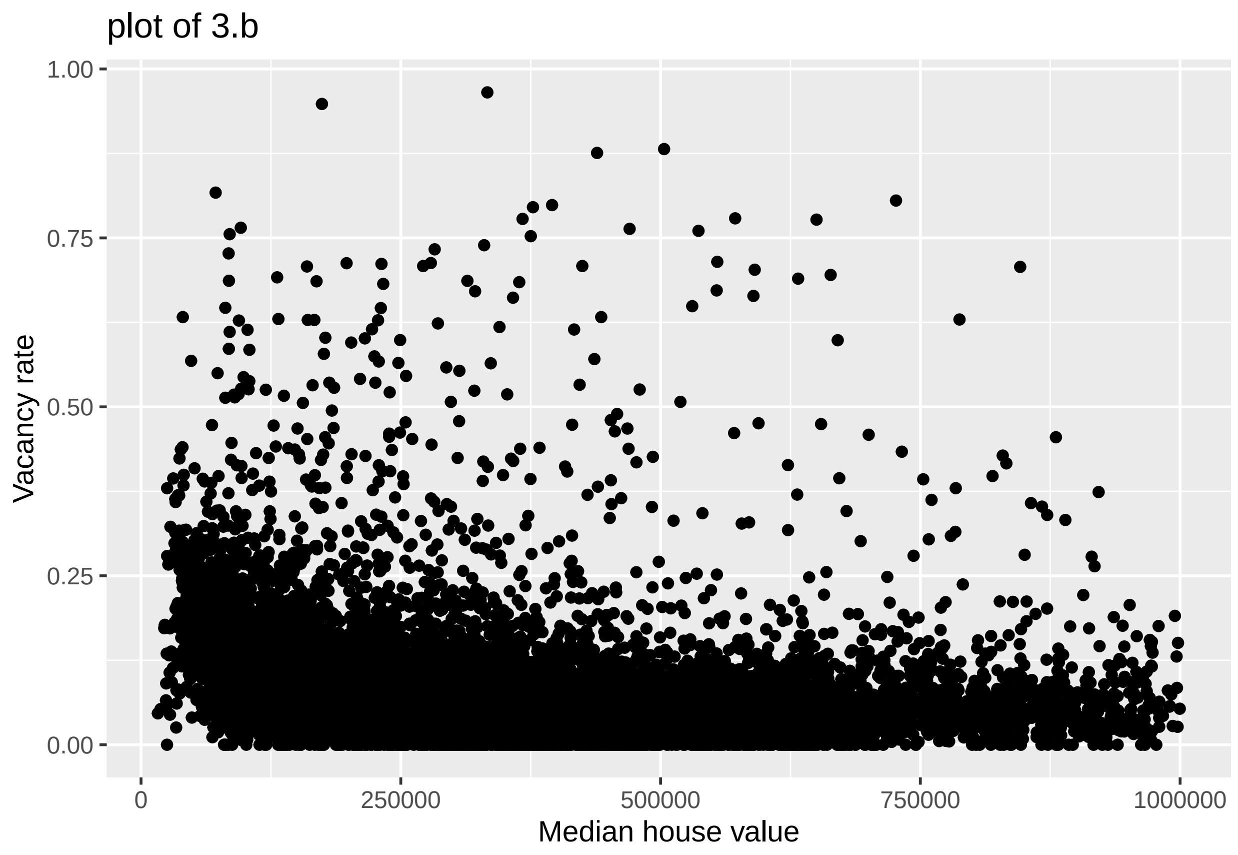
```
summary(ca_pa$Vacancy_rate)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.03846 0.06767 0.08889 0.10921 0.96531
```

A new column named Vacancy_rate indicating just the vacancy rate has been added into the ca_pa data frame, and according to the information the method summary spit out the minimum, maximum, mean, and median vacancy rates are 0.00000, 0.96531, 0.08889, 0.06767, respectively.

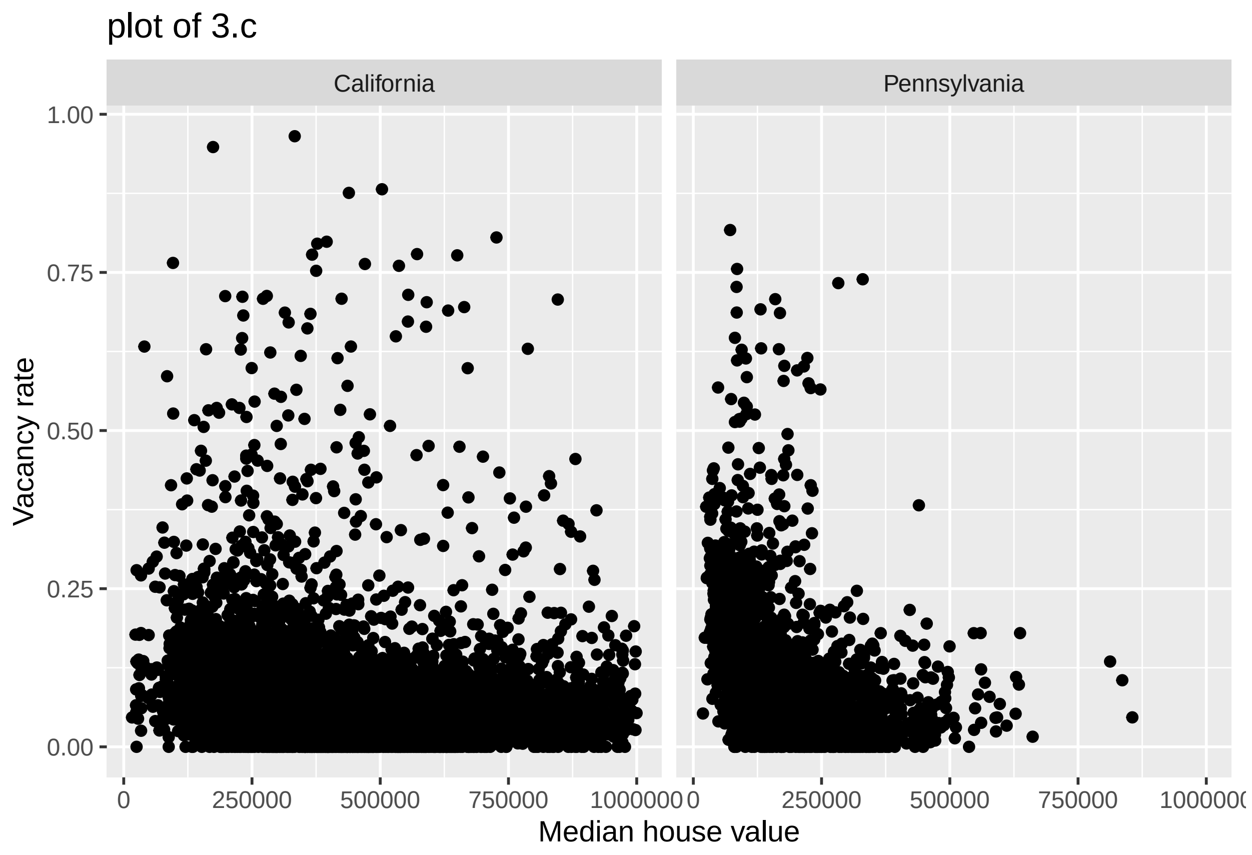
**b. Plot the vacancy rate against median house value.**

```
ca_pa %>% ggplot() +
  geom_point(aes(x = Median_house_value, y = Vacancy_rate)) +
  labs(x = "Median house value", y = "Vacancy rate", title = "plot of 3.b")
```



**c. Plot vacancy rate against median house value separately for California and for Pennsylvania. Is there a difference?**

```
ca_pa %>% ggplot() +
  geom_point(aes(x = Median_house_value, y = Vacancy_rate)) +
  labs(x = "Median house value", y = "Vacancy rate", title = "plot of 3.c") +
  facet_wrap(~STATEFP, labeller = as_labeller(names))
```



So in this pair of plots one can see that there is indeed a difference between the two state as the vacancy rate spread out evenly over median house value in California but high vacancy rate mainly occurs at low median house value in Pennsylvania.

## 4

**a. Explain what the block of code at the end of this question is supposed to accomplish, and how it does it.**

```
# acca <- c()
# for (tract in 1:nrow(ca_pa)) {
#   if (ca_pa$STATEFP[tract] == 6) {
#     if (ca_pa$COUNTYFP[tract] == 1) {
#       acca <- c(acca, tract)
#     }
#   }
# }
# accamhv <- c()
# for (tract in acca) {
#   accamhv <- c(accamhv, ca_pa[tract,10])
# }
# median(accamhv)
```

The code above is try to get the median of the median house value of all. It first use a for loop to record the number of rows where STATEFP is 6 and COUNTYFP 1 and store them into the variable acca, then in the for loop below using it to grab out the 10th column of the corresponding rows, which is the Median_house_value, and stores it into the variable accamhv, and finally do the median upon it.

**b. Give a single line of R which gives the same final answer as the block of code. Note: there are at least two ways to do this; you just have to find one.**

```
ca_pa %>% filter(STATEFP == "06" & COUNTYFP == "001") %>%
  dplyr::select(Median_house_value) %>% apply(2, median)
```

```
## Median_house_value
##             474050
```

**c. For Alameda, Santa Clara and Allegheny Counties, what were the average percentages of housing built since 2005?**

```
# colnames(ca_pa)
ca_pa.mean.Built_2005_or_later <- ca_pa %>% group_by(STATEFP, COUNTYFP) %>%
  summarise(Mean_Built_2005_or_later = mean(Built_2005_or_later)) %>% ungroup()
```

```
## `summarise()` has grouped output by 'STATEFP'. You can override using the `.groups` argument.
```

```
ca_pa.mean.Built_2005_or_later %>% filter(STATEFP == '06') %>%
  filter(COUNTYFP %in% c('001', '085'))
```

```
## # A tibble: 2 x 3
##   STATEFP COUNTYFP Mean_Built_2005_or_later
##   <chr>   <chr>                       <dbl>
## 1 06      001                          2.82
## 2 06      085                          3.20
```

```
ca_pa.mean.Built_2005_or_later %>% filter(STATEFP == '42') %>%
  filter(COUNTYFP %in% c('003'))
```

```
## # A tibble: 1 x 3
##   STATEFP COUNTYFP Mean_Built_2005_or_later
##   <chr>   <chr>                       <dbl>
## 1 42      003                          1.47
```

The the average percentages of housing built since 2005 of Alameda, Santa Clara and Allegheny Counties are 2.820468, 3.200319 and 1.474219, respectively.

**d. The cor function calculates the correlation coefficient between two variables. What is the correlation between median house value and the percent of housing built since 2005 in (i) the whole data, (ii) all of California, (iii) all of Pennsylvania, (iv) Alameda County, (v) Santa Clara County and (vi) Allegheny County?**

(i)

```
ca_pa %>% dplyr::select(Median_house_value, Built_2005_or_later) %>% cor()
```

```
##                     Median_house_value Built_2005_or_later
## Median_house_value          1.00000000         -0.01893186
## Built_2005_or_later        -0.01893186          1.00000000
```

(ii)

```
ca_pa %>% filter(STATEFP == '06') %>%
  dplyr::select(Median_house_value, Built_2005_or_later) %>%cor()
```

```
##                     Median_house_value Built_2005_or_later
## Median_house_value           1.0000000          -0.1153604
## Built_2005_or_later         -0.1153604           1.0000000
```

(iii)

```
ca_pa %>% filter(STATEFP == '42') %>%
  dplyr::select(Median_house_value, Built_2005_or_later) %>%cor()
```

```
##                     Median_house_value Built_2005_or_later
## Median_house_value           1.0000000           0.2681654
## Built_2005_or_later          0.2681654           1.0000000
```

(iv)

```
ca_pa %>% filter(STATEFP == '06' & COUNTYFP == '001') %>%
  dplyr::select(Median_house_value, Built_2005_or_later) %>%cor()
```

```
##                     Median_house_value Built_2005_or_later
## Median_house_value          1.00000000          0.01303543
## Built_2005_or_later         0.01303543          1.00000000
```

(v)

```
ca_pa %>% filter(STATEFP == '06' & COUNTYFP == '085') %>%
  dplyr::select(Median_house_value, Built_2005_or_later) %>%cor()
```

```
##                     Median_house_value Built_2005_or_later
## Median_house_value           1.0000000          -0.1726203
## Built_2005_or_later         -0.1726203           1.0000000
```

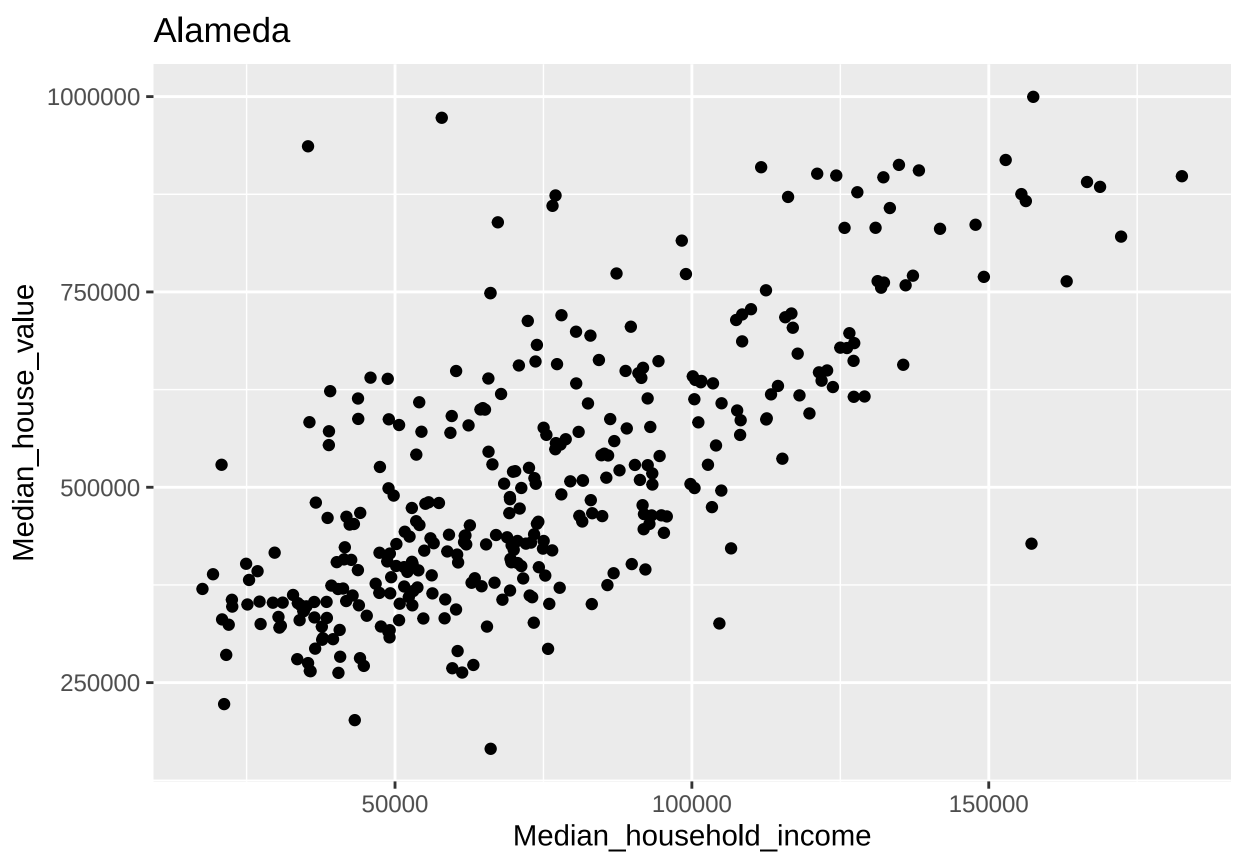
(vi)

```
ca_pa %>% filter(STATEFP == '42' & COUNTYFP == '003') %>%
  dplyr::select(Median_house_value, Built_2005_or_later) %>%cor()
```

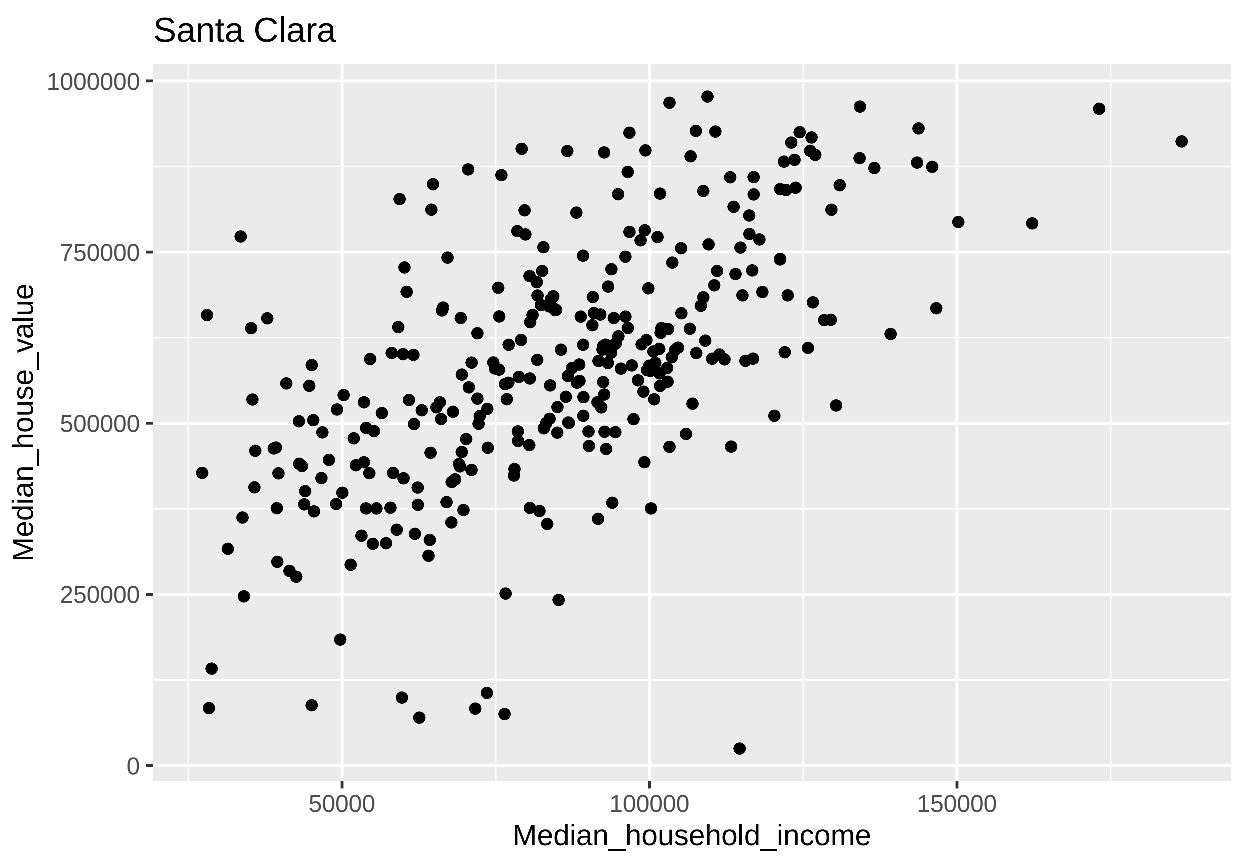
```
##                     Median_house_value Built_2005_or_later
## Median_house_value           1.0000000           0.1939652
## Built_2005_or_later          0.1939652           1.0000000
```

**e. Make three plots, showing median house values against median income, for Alameda, Santa Clara, and Allegheny Counties. (If you can fit the information into one plot, clearly distinguishing the three counties, that's OK too.)**

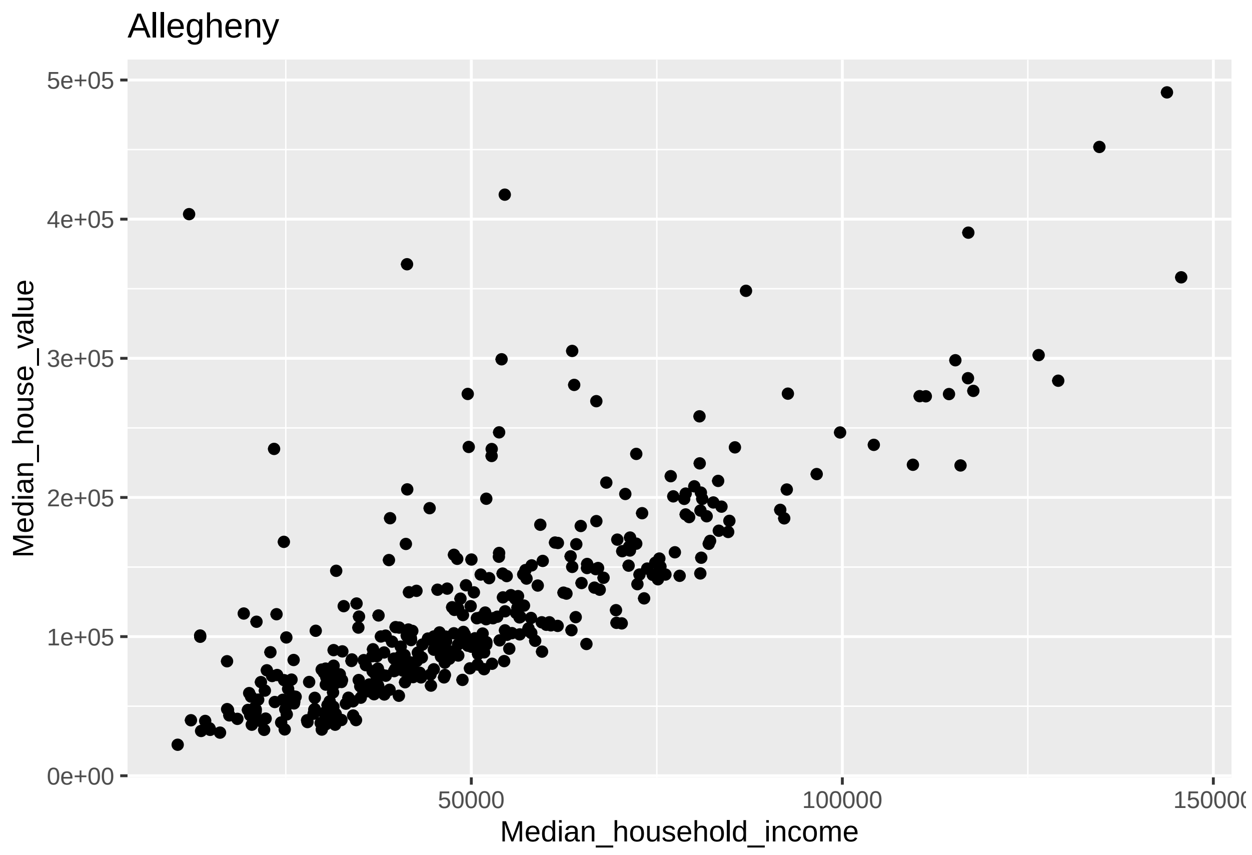
```
ca_pa.4e <- ca_pa %>%
  dplyr::select(STATEFP, COUNTYFP, Median_household_income, Median_house_value)
ca_pa.4e %>% filter(STATEFP == '06' & COUNTYFP == '001') %>% ggplot() +
  geom_point(aes(x = Median_household_income, y = Median_house_value)) +
  labs(title = 'Alameda')
```



```
ca_pa.4e %>% filter(STATEFP == '06' & COUNTYFP == '085') %>% ggplot() +
  geom_point(aes(x = Median_household_income, y = Median_house_value)) +
  labs(title = 'Santa Clara')
```



```
ca_pa.4e %>% filter(STATEFP == '42' & COUNTYFP == '003') %>% ggplot() +
  geom_point(aes(x = Median_household_income, y = Median_house_value)) +
  labs(title = "Allegheny")
```



## MB.Ch1.11

```
# gender <- factor(c(rep("female", 91), rep("male", 92)))
# table(gender)
```

1. The factor gender is created with female in the first place and the output shows just what the gender factor supposes to be.

```
# gender <- factor(gender, levels=c("male", "female"))
# table(gender)
```

2. The first line of the code redefined the order of the factors inside the gender factor, which is now male comes the first, and the output of method table is changed, correspondingly.

```
# gender <- factor(gender, levels=c("Male", "female"))
# # Note the mistake: "Male" should be "male"
# table(gender)
```

3. This time the first line of the code gave the factor gender a new factor named Male and which is not exist inside gender, as showed in the output table, in which the number of factor Male is exactly 0.

```
# table(gender, exclude=NULL)
```

4. This new method table code line added an additional parameter and therefore finds the origin male factors back.

## MB.Ch1.12

**(a) Use the sequence of numbers 1, 2, . . . , 100 to check that this function gives the result that is expected.**

The function is:

```
# input: the vector x and the value cutoff v
# return: the proportion
my_func <- function(x, v){
  return(mean(x > v))
}
```

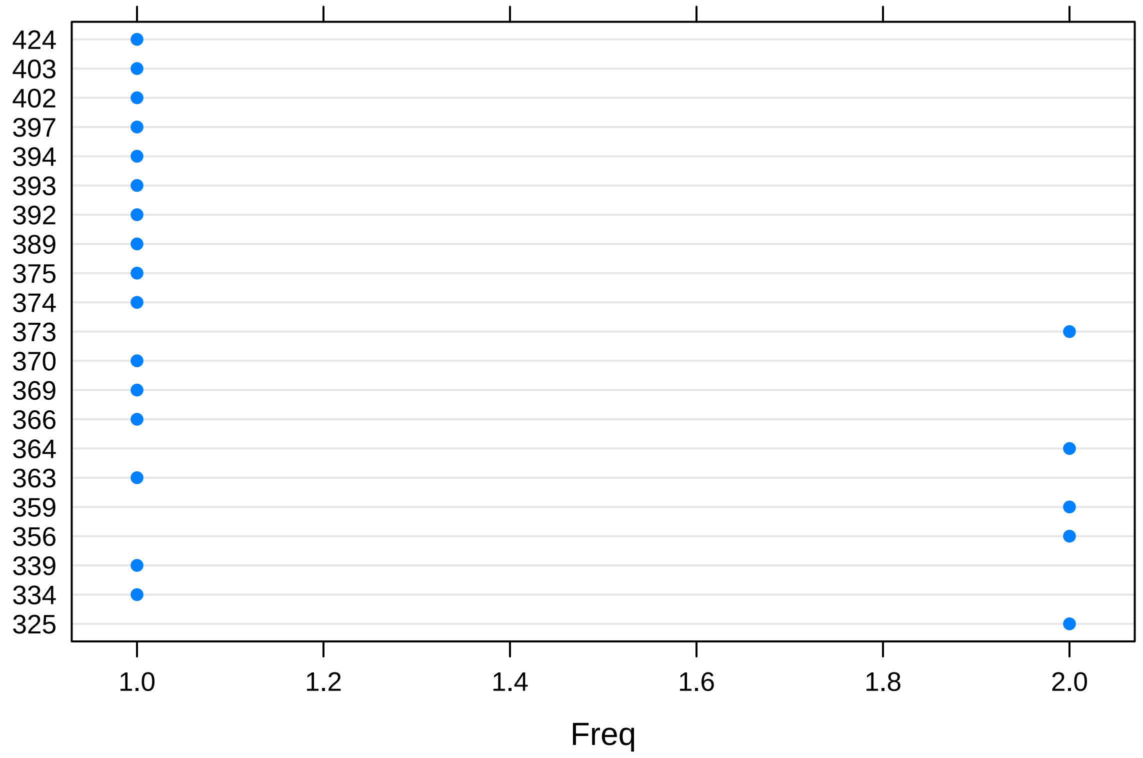
```
my_func(1 : 100, 50)
```

```
## [1] 0.5
```

which is exactly the proportion of values in the vector bigger than 50.

**(b) Obtain the vector ex01.36 from the Devore6 (or Devore7) package. These data give the times required for individuals to escape from an oil platform during a drill. Use dotplot() to show the distribution of times. Calculate the proportion of escape times that exceed 7 minutes.**

```
# ex01.36
dotplot(ex01.36)
```



```
my_func(ex01.36, 7 * 60)
```

```
## [1] 0.03846154
```

## MB.Ch1.18

```
Treatment <- unstack(Rabbit, Treatment ~ Animal)[[1]]
Dose <- unstack(Rabbit, Dose ~ Animal)[[1]]
BPchange <- unstack(Rabbit, BPchange ~ Animal)
Rabbit_new <- data.frame(Treatment, Dose, BPchange)
Rabbit_new
```

```
##    Treatment   Dose    R1    R2    R3    R4   R5
## 1    Control   6.25  0.50  1.00  0.75  1.25  1.5
## 2    Control  12.50  4.50  1.25  3.00  1.50  1.5
## 3    Control  25.00 10.00  4.00  3.00  6.00  5.0
## 4    Control  50.00 26.00 12.00 14.00 19.00 16.0
## 5    Control 100.00 37.00 27.00 22.00 33.00 20.0
## 6    Control 200.00 32.00 29.00 24.00 33.00 18.0
## 7        MDL   6.25  1.25  1.40  0.75  2.60  2.4
## 8        MDL  12.50  0.75  1.70  2.30  1.20  2.5
## 9        MDL  25.00  4.00  1.00  3.00  2.00  1.5
## 10       MDL  50.00  9.00  2.00  5.00  3.00  2.0
## 11       MDL 100.00 25.00 15.00 26.00 11.00  9.0
## 12       MDL 200.00 37.00 28.00 25.00 22.00 19.0
```