# STAT 206 Homework 2

**Due Monday, October 16, 5:00 PM**

***General instructions for homework***: Homework must be completed as an R Markdown file. Be sure to include your name in the file. Give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Each answer must be supported by written statements as well as any code used. (Examining your various objects in the "Environment" section of RStudio is insufficient – you must use scripted commands.)

The data set at [http://www.stat.cmu.edu/~cshalizi/uADA/13/hw/01/calif_penn_2011.csv] contains information about the housing stock of California and Pennsylvania, as of 2011. Information as aggregated into "Census tracts", geographic regions of a few thousand people which are supposed to be fairly homogeneous economically and socially.

1. *Loading and cleaning*
    a. Load the data into a dataframe called `ca_pa`.

```
#Since this is a csv file. We will use the read.csv functions instead of read.table like last week
#To read the table I am placing the URL in its own variable. Also I am placing the data in an object na

WWW = "http://www.stat.cmu.edu/~cshalizi/uADA/13/hw/01/calif_penn_2011.csv"
ca_pa = read.csv(WWW)
class(data)
```

```
## [1] "function"
```

```
head(data, 370)
```

```
##
## 1   function (..., list = character(), package = NULL, lib.loc = NULL,
## 2       verbose = getOption("verbose"), envir = .GlobalEnv)
## 3   {
## 4       fileExt <- function(x) {
## 5           db <- grepl("\\\\.[^.]+\\\\.(gz|bz2|xz)$", x)
## 6           ans <- sub(".*\\\\.", "", x)
## 7           ans[db] <- sub(".*\\\\.([^.]+\\\\.)(gz|bz2|xz)$", "\\\\1\\\\2",
## 8               x[db])
## 9           ans
## 10      }
## 11      names <- c(as.character(substitute(list(...))[-1L]), list)
## 12      if (!is.null(package)) {
## 13          if (!is.character(package))
## 14              stop("'package' must be a character string or NULL")
## 15          if (any(package %in% "base"))
## 16              warning("datasets have been moved from package 'base' to package 'datasets'")
## 17          if (any(package %in% "stats"))
## 18              warning("datasets have been moved from package 'stats' to package 'datasets'")
## 19          package[package %in% c("base", "stats")] <- "datasets"
## 20      }
## 21      paths <- find.package(package, lib.loc, verbose = verbose)
## 22      if (is.null(lib.loc))
## 23          paths <- c(path.package(package, TRUE), if (!length(package)) getwd(),
## 24              paths)
## 25      paths <- unique(normalizePath(paths[file.exists(paths)]))
```

```
## 26      paths <- paths[dir.exists(file.path(paths, "data"))]
## 27      dataExts <- tools:::.make_file_exts("data")
## 28      if (length(names) == 0L) {
## 29          db <- matrix(character(), nrow = 0L, ncol = 4L)
## 30          for (path in paths) {
## 31              entries <- NULL
## 32              packageName <- if (file_test("-f", file.path(path,
## 33                  "DESCRIPTION")))
## 34                  basename(path)
## 35              else "."
## 36              if (file_test("-f", INDEX <- file.path(path, "Meta",
## 37                  "data.rds"))) {
## 38                  entries <- readRDS(INDEX)
## 39              }
## 40              else {
## 41                  dataDir <- file.path(path, "data")
## 42                  entries <- tools::list_files_with_type(dataDir,
## 43                    "data")
## 44                  if (length(entries)) {
## 45                    entries <- unique(tools::file_path_sans_ext(basename(entries)))
## 46                    entries <- cbind(entries, "")
## 47                  }
## 48              }
## 49              if (NROW(entries)) {
## 50                  if (is.matrix(entries) && ncol(entries) == 2L)
## 51                    db <- rbind(db, cbind(packageName, dirname(path),
## 52                      entries))
## 53                  else warning(gettextf("data index for package %s is invalid and will be ignored"
## 54                    sQuote(packageName)), domain = NA, call. = FALSE)
## 55              }
## 56          }
## 57          colnames(db) <- c("Package", "LibPath", "Item", "Title")
## 58          footer <- if (missing(package))
## 59              paste0("Use ", sQuote(paste("data(package =", ".packages(all.available = TRUE))")),
## 60                  "\\n", "to list the data sets in all *available* packages.")
## 61          else NULL
## 62          y <- list(title = "Data sets", header = NULL, results = db,
## 63              footer = footer)
## 64          class(y) <- "packageIQR"
## 65          return(y)
## 66      }
## 67      paths <- file.path(paths, "data")
## 68      for (name in names) {
## 69          found <- FALSE
## 70          for (p in paths) {
## 71              if (file_test("-f", file.path(p, "Rdata.rds"))) {
## 72                  rds <- readRDS(file.path(p, "Rdata.rds"))
## 73                  if (name %in% names(rds)) {
## 74                    found <- TRUE
## 75                    if (verbose)
## 76                      message(sprintf("name=%s:\\t found in Rdata.rds",
## 77                        name), domain = NA)
## 78                    thispkg <- sub(".*/([^/]*)/data$", "\\\\1", p)
## 79                    thispkg <- sub("_.*$", "", thispkg)
```

```
## 80                    thispkg <- paste0("package:", thispkg)
## 81                    objs <- rds[[name]]
## 82                    lazyLoad(file.path(p, "Rdata"), envir = envir,
## 83                      filter = function(x) x %in% objs)
## 84                    break
## 85                  }
## 86                  else if (verbose)
## 87                    message(sprintf("name=%s:\\t NOT found in names() of Rdata.rds, i.e.,\\n\\t%s\
## 88                      name, paste(names(rds), collapse = ",")),
## 89                      domain = NA)
## 90              }
## 91              if (file_test("-f", file.path(p, "Rdata.zip"))) {
## 92                  warning("zipped data found for package ", sQuote(basename(dirname(p))),
## 93                    ".\\nThat is defunct, so please re-install the package.",
## 94                    domain = NA)
## 95                  if (file_test("-f", fp <- file.path(p, "filelist")))
## 96                    files <- file.path(p, scan(fp, what = "", quiet = TRUE))
## 97                  else {
## 98                    warning(gettextf("file 'filelist' is missing for directory %s",
## 99                      sQuote(p)), domain = NA)
## 100                   next
## 101                 }
## 102             }
## 103             else {
## 104                 files <- list.files(p, full.names = TRUE)
## 105             }
## 106             files <- files[grep(name, files, fixed = TRUE)]
## 107             if (length(files) > 1L) {
## 108                 o <- match(fileExt(files), dataExts, nomatch = 100L)
## 109                 paths0 <- dirname(files)
## 110                 paths0 <- factor(paths0, levels = unique(paths0))
## 111                 files <- files[order(paths0, o)]
## 112             }
## 113             if (length(files)) {
## 114                 for (file in files) {
## 115                   if (verbose)
## 116                     message("name=", name, ":\\t file= ...", .Platform$file.sep,
## 117                       basename(file), "::\\t", appendLF = FALSE,
## 118                       domain = NA)
## 119                   ext <- fileExt(file)
## 120                   if (basename(file) != paste0(name, ".", ext))
## 121                     found <- FALSE
## 122                   else {
## 123                     found <- TRUE
## 124                     zfile <- file
## 125                     zipname <- file.path(dirname(file), "Rdata.zip")
## 126                     if (file.exists(zipname)) {
## 127                       Rdatadir <- tempfile("Rdata")
## 128                       dir.create(Rdatadir, showWarnings = FALSE)
## 129                       topic <- basename(file)
## 130                       rc <- .External(C_unzip, zipname, topic,
## 131                         Rdatadir, FALSE, TRUE, FALSE, FALSE)
## 132                       if (rc == 0L)
## 133                         zfile <- file.path(Rdatadir, topic)
```

```
## 134                     }
## 135                     if (zfile != file)
## 136                       on.exit(unlink(zfile))
## 137                     switch(ext, R = , r = {
## 138                       library("utils")
## 139                       sys.source(zfile, chdir = TRUE, envir = envir)
## 140                     }, RData = , rdata = , rda = load(zfile,
## 141                       envir = envir), TXT = , txt = , tab = ,
## 142                       tab.gz = , tab.bz2 = , tab.xz = , txt.gz = ,
## 143                       txt.bz2 = , txt.xz = assign(name, read.table(zfile,
## 144                         header = TRUE, as.is = FALSE), envir = envir),
## 145                       CSV = , csv = , csv.gz = , csv.bz2 = ,
## 146                       csv.xz = assign(name, read.table(zfile,
## 147                         header = TRUE, sep = ";", as.is = FALSE),
## 148                         envir = envir), found <- FALSE)
## 149                   }
## 150                   if (found)
## 151                     break
## 152                 }
## 153                 if (verbose)
## 154                   message(if (!found)
## 155                     "*NOT* ", "found", domain = NA)
## 156             }
## 157             if (found)
## 158                 break
## 159         }
## 160         if (!found)
## 161             warning(gettextf("data set %s not found", sQuote(name)),
## 162                 domain = NA)
## 163     }
## 164     invisible(names)
## 165 }
```

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

```
# to find the number of columns and rows we will use the nrow and ncol function like we did in the lab.
# number of columns is also displayed in the previous cell at the bottom of the head function used

nrow(ca_pa)
```

```
## [1] 11275
```

```
ncol(ca_pa)
```

```
## [1] 34
```

c. Run this command, and explain, in words, what this does:

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

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

```
##                          X                     GEO.id2
##                          0                           0
##                    STATEFP                    COUNTYFP
##                          0                           0
##                    TRACTCE                  POPULATION
##                          0                           0
```

```
##                     LATITUDE                    LONGITUDE
##                            0                            0
##            GEO.display.label           Median_house_value
##                            0                          599
##                 Total_units                Vacant_units
##                            0                            0
##                Median_rooms  Mean_household_size_owners
##                          157                          215
## Mean_household_size_renters         Built_2005_or_later
##                          152                           98
##           Built_2000_to_2004                 Built_1990s
##                           98                           98
##                 Built_1980s                 Built_1970s
##                           98                           98
##                 Built_1960s                 Built_1950s
##                           98                           98
##                 Built_1940s       Built_1939_or_earlier
##                           98                           98
##                  Bedrooms_0                  Bedrooms_1
##                           98                           98
##                  Bedrooms_2                  Bedrooms_3
##                           98                           98
##                  Bedrooms_4          Bedrooms_5_or_more
##                           98                           98
##                      Owners                      Renters
##                          100                          100
##     Median_household_income       Mean_household_income
##                          115                          126
```

```
#The outside function is to get a sum of something which will be determined by the given arguments
#The apply is used for iteration, the first argument for apply is the dataframe
#The second argument specifies that the apply will be used on rows and columns
#The third argument is the function, or what we want to get a sum of in this case
#in other words we are counting the number of N/As per row and column in the data frame.
```

d. The function `na.omit()` takes a dataframe and returns a new dataframe, omitting any row containing a

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

e. How many rows did this eliminate?

```
# Above we can see that it says 3,575 rows. We originally had 3876. We can use r as a calculator for th
3876-3575
```

```
## [1] 301
```

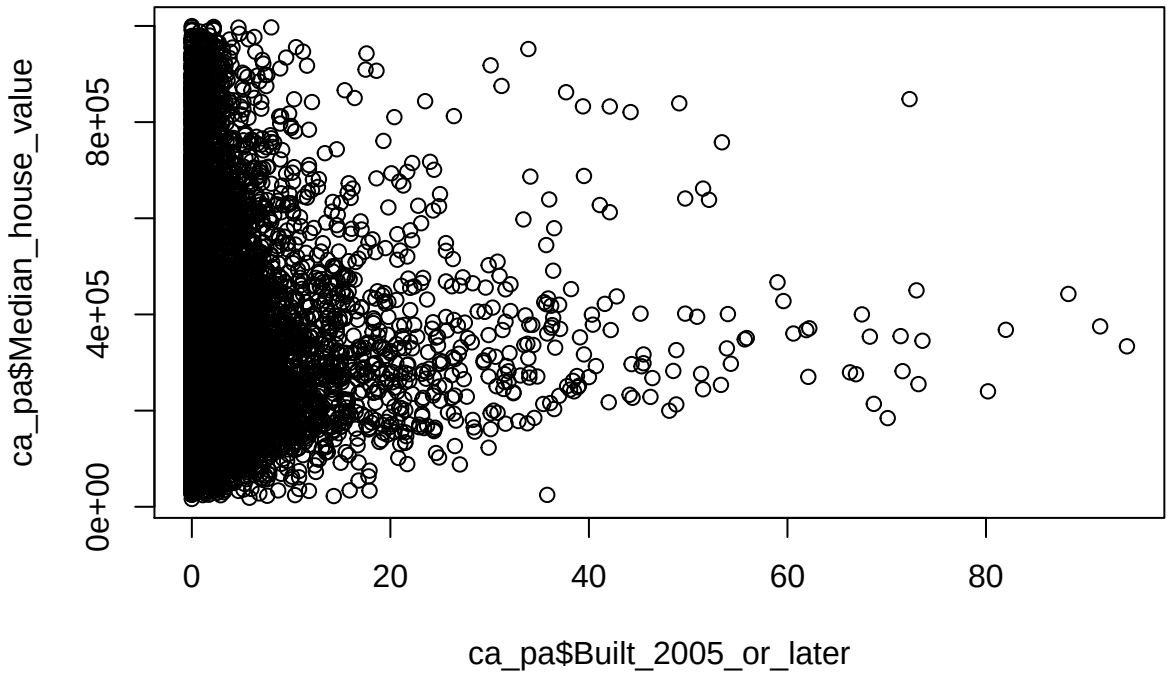
f. Are your answers in (c) and (e) compatible? Explain.

```
# Not quite, in (c) we are getting the counts of na per attribute. There can be multiple na values per
# So the total number of na in the data frame is not going to equal the number of rows that are removed
# by the omit.na function
```

2. *This Very New House*
   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.

```
#using the plot function the first argument will be Built_2005_or_Later and the second for the y axis i

plot(ca_pa$Built_2005_or_later, ca_pa$Median_house_value)
```




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.

```
# I am going to make two subsets of data to makes these plots by state, one for ca and one for pa

class(ca_pa)
```

```
## [1] "data.frame"
```

```
names(ca_pa)
```

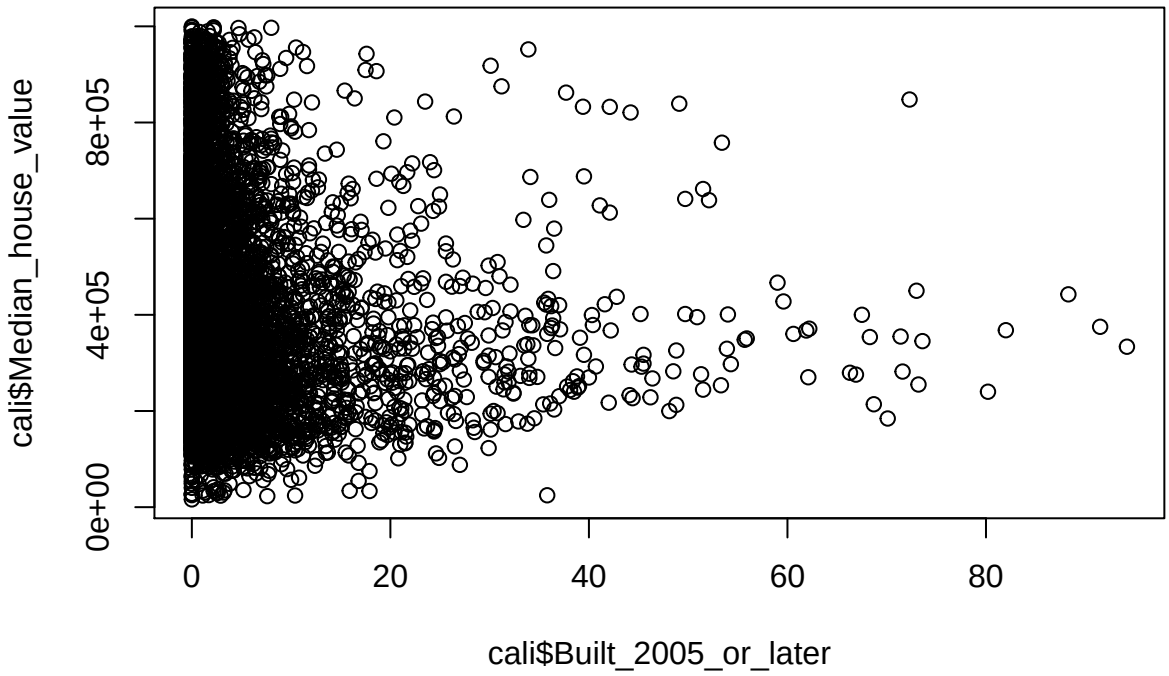
```
##  [1] "X"                                "GEO.id2"
##  [3] "STATEFP"                          "COUNTYFP"
##  [5] "TRACTCE"                          "POPULATION"
##  [7] "LATITUDE"                         "LONGITUDE"
##  [9] "GEO.display.label"                "Median_house_value"
## [11] "Total_units"                      "Vacant_units"
## [13] "Median_rooms"                     "Mean_household_size_owners"
## [15] "Mean_household_size_renters"      "Built_2005_or_later"
## [17] "Built_2000_to_2004"               "Built_1990s"
## [19] "Built_1980s"                      "Built_1970s"
## [21] "Built_1960s"                      "Built_1950s"
## [23] "Built_1940s"                      "Built_1939_or_earlier"
## [25] "Bedrooms_0"                       "Bedrooms_1"
```

```
## [27] "Bedrooms_2"                  "Bedrooms_3"
## [29] "Bedrooms_4"                  "Bedrooms_5_or_more"
## [31] "Owners"                      "Renters"
## [33] "Median_household_income"     "Mean_household_income"
```

```
cali = subset(ca_pa, STATEFP == 6, select = c('Built_2005_or_later', 'Median_house_value'))
nrow(cali)
```

```
## [1] 7481
```

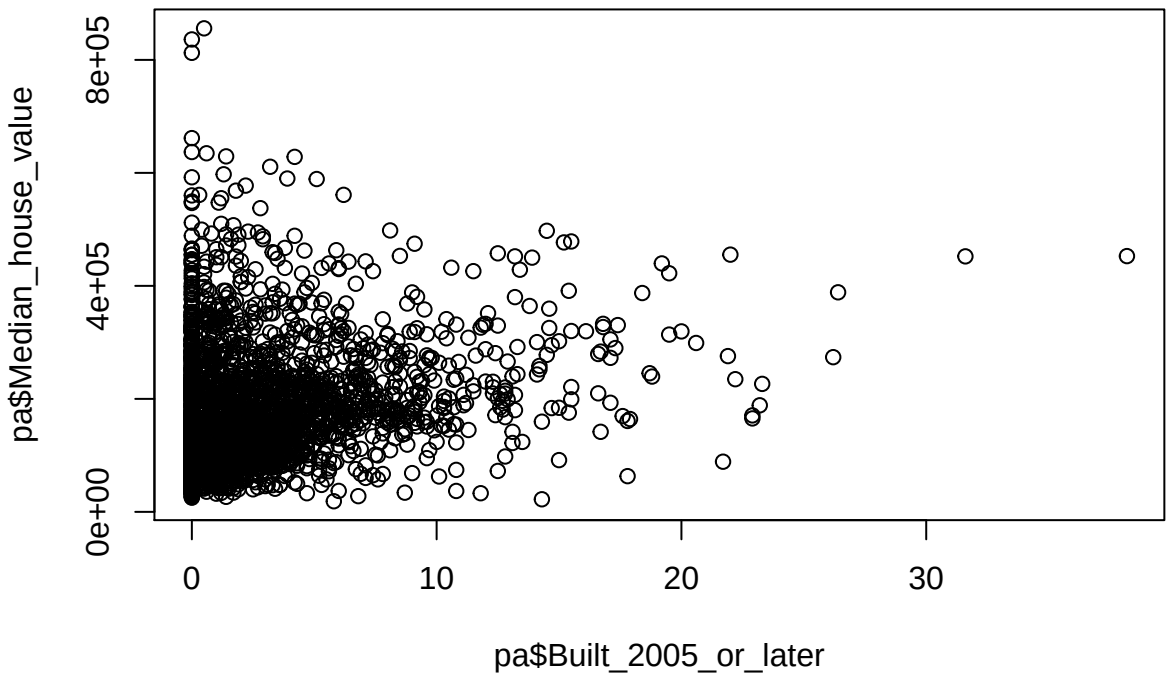
```
plot(cali$Built_2005_or_later, cali$Median_house_value)
```




```
pa = subset(ca_pa, STATEFP == 42)
nrow(pa)
```

```
## [1] 3124
```

```
plot(pa$Built_2005_or_later, pa$Median_house_value)
```




3. *Nobody Home*
   The vacancy rate is the fraction of housing units which are not occupied. The dataframe contains columns giving the total number of housing units for each Census tract, and the number of vacant housing units.
   a. Add a new column to the dataframe which contains the vacancy rate. What are the minimum, maximum, mean, and median vacancy rates?

```
#we are going to use the transform function to add another column tothe ca_pa dataframe which will be
#the ratio of vacant units to total units
#finally we will get a summary of that column for all the values we want

ca_pa_new = transform(ca_pa, vac_rat = ca_pa$Vacant_units / ca_pa$Total_units)
summary(ca_pa_new$vac_rat)
```

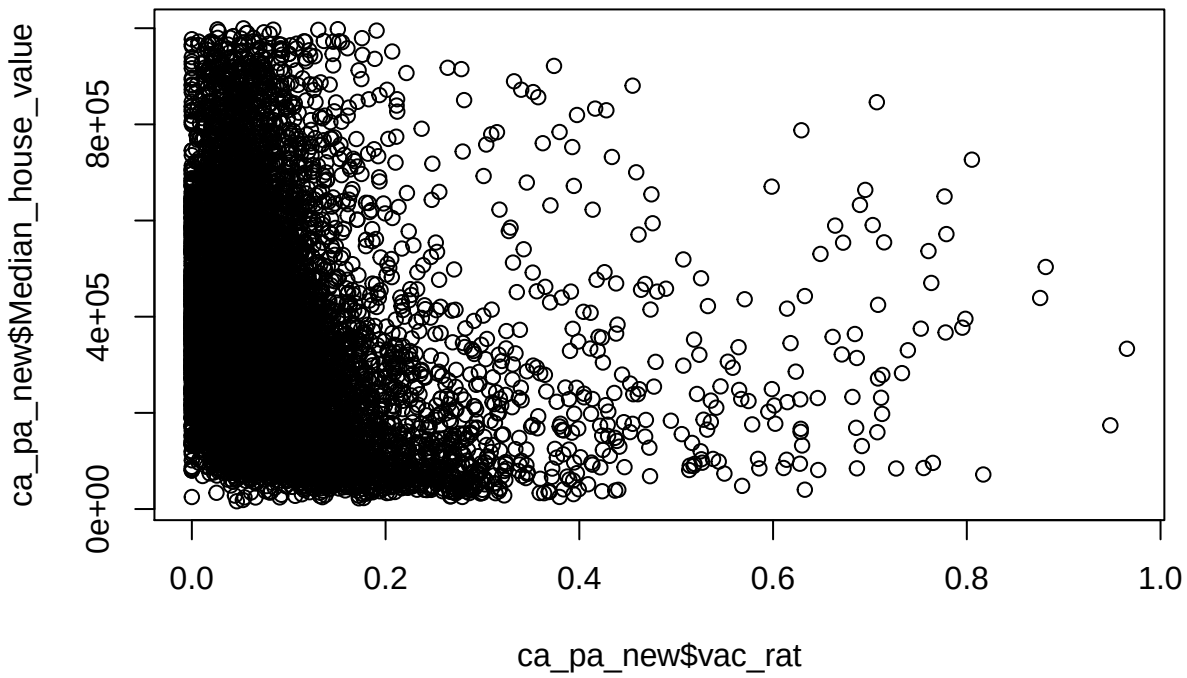
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.03846 0.06767 0.08889 0.10921 0.96531
```

```
#minimum: 0
#max : 1
#mean: .08918
#median: .06766
```

b. Plot the vacancy rate against median house value.

```
#again using the plot function but this time drawing from the new data frame with the vac_rat column

plot(ca_pa_new$vac_rat, ca_pa_new$Median_house_value)
```

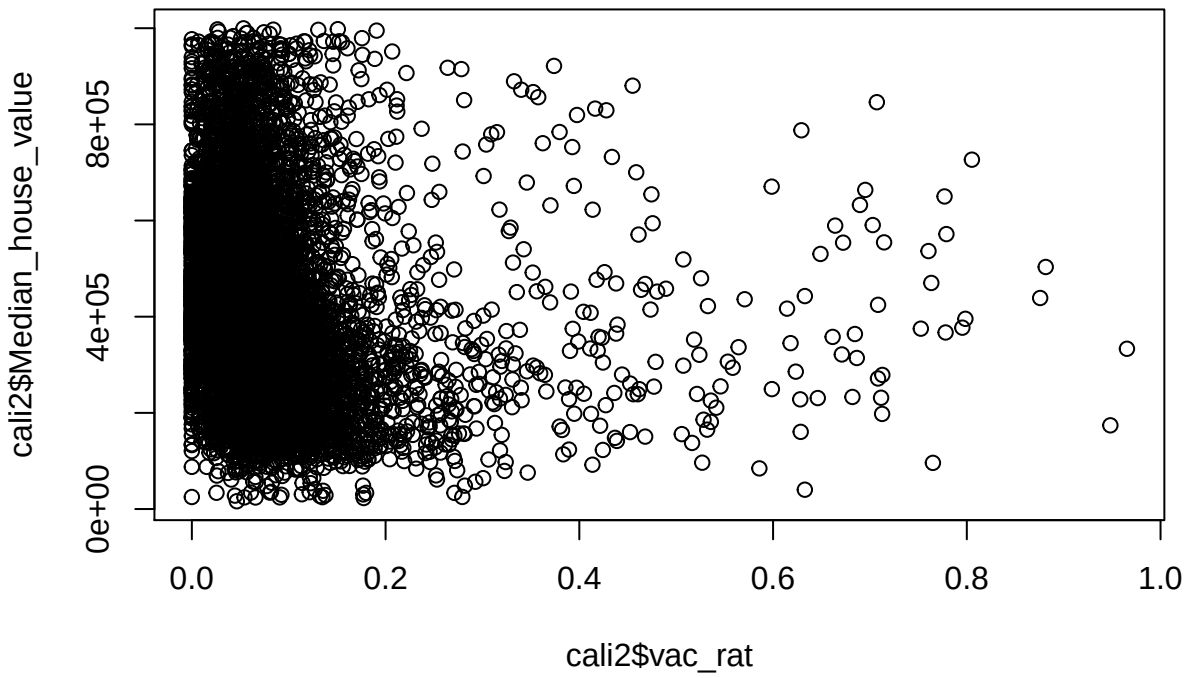



c. Plot vacancy rate against median house value separately for California and for Pennsylvania. Is there a difference?

```
cali2 = subset(ca_pa_new, STATEFP == 6, select = c('vac_rat', 'Median_house_value'))
nrow(cali2)
```

```
## [1] 7481
```

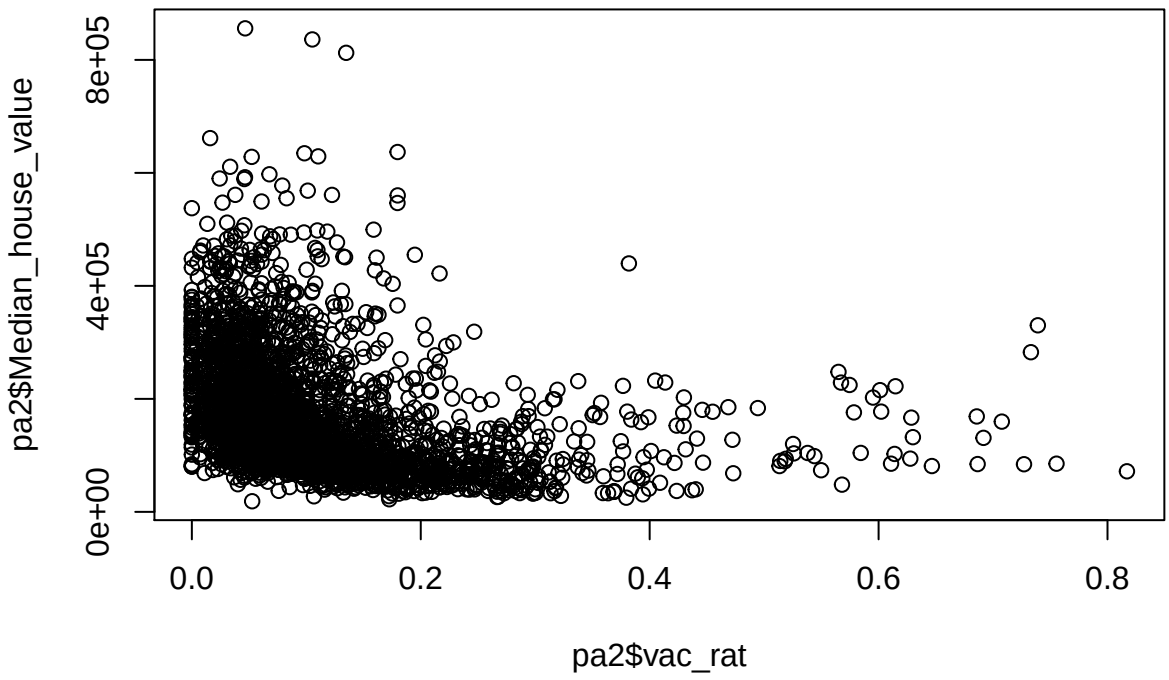
```
plot(cali2$vac_rat, cali2$Median_house_value)
```




```
pa2 = subset(ca_pa_new, STATEFP == 42, select = c('vac_rat', 'Median_house_value'))
nrow(pa2)
```

```
## [1] 3124
```

```
plot(pa2$vac_rat, pa2$Median_house_value)
```




4. The column `COUNTYFP` contains a numerical code for counties within each state. We are interested in Alameda County (county 1 in California), Santa Clara (county 85 in California), and Allegheny County (county 3 in Pennsylvania).
   a. Explain what the block of code at the end of this question is supposed to accomplish, and how it does it.

```
#The code ultimatly gets the average median house value for lines items that are in california AND alam
#
#
#line by line explanation:
#The first line sets an empty vector to the object name "acca"
#The seccond line begins a for loop that iterates through the dataframe beginning at the first row and
#to the last, line by line
#The next line is an if statement to figure out if the STATEFP has a value of 6, if so it will continue
#next if statement, if not it will restart the for loop (looking for california)
#The next if statement determines if the COUNTYFP is 1 (looking for alameda county)
#If the above row number is saved in the vector acca
#Next the code creates a vector called accamhv
#The second for loop goes through each value in the value in acca and looks up the value in the 10th co
#this value is the median house value and it is saved to a vector of length = to length of acca
#NOTE some values are NA which ultimately ends up being spit out of the median function, These NA value
#to be handled to get a numeric value out
```

b. Give a single line of R which gives the same final answer as the block of code. Note: there are at l

```
#This code calls median first and the argument passed is to grab the Median house value column and furt
#COUNTYFP equal to 1
```

```r
median(ca_pa$Median_house_value['COUNTYFP' == 1])
```

```
## [1] NA
```

c. For Alameda, Santa Clara and Allegheny Counties, what were the average percentages of housing built s

```r
#For this section we just need to slice the data so we get the housing built since 2005 and the county
#then we have the percentages of houses built for those 3 countys and we can find the avg for each coun

alameda = subset(ca_pa, COUNTYFP == 1, select = c('COUNTYFP', 'Built_2005_or_later'))
alameda2 = na.omit(alameda)
mean(alameda2$Built_2005_or_later)
```

```
## [1] 3.029589
```

```r
santa = subset(ca_pa, COUNTYFP == 85, select = c('COUNTYFP', 'Built_2005_or_later'))
santa2 = na.omit(santa)
mean(santa2$Built_2005_or_later)
```

```
## [1] 3.072595
```

```r
alle = subset(ca_pa, COUNTYFP == 3, select = c('COUNTYFP', 'Built_2005_or_later'))
alle2 = na.omit(alle)
mean(alle2$Built_2005_or_later)
```

```
## [1] 1.475844
```

d. The `cor` function calculates the correlation coefficient between two variables.  What is the correla

```r
#correlation between median house value and perfect of housing build since 2005
#To use the cor function for these attributes we need to handle non-numeric values
#For this we use the optional argument use = "complete.obs"

#the whole data

cor(ca_pa$Median_house_value, ca_pa$Built_2005_or_later, use = "complete.obs", method = "pearson")
```

```
## [1] -0.01893186
```

```r
#all california

calsub = subset(ca_pa, STATEFP == 6, select = c('Median_house_value', 'Built_2005_or_later'))
#calsub

cor(calsub, use = "complete.obs", method = "pearson")
```

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

```r
#all of Pennsylvania

pasub = subset(ca_pa, STATEFP == 42, select = c('Median_house_value', 'Built_2005_or_later'))
#pasub

cor(pasub, use = "complete.obs", method = "pearson")
```

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

```
## Built_2005_or_later          0.2681654           1.0000000
```

```
#Alameda county

acsub = subset(ca_pa, COUNTYFP == 1, select = c('Median_house_value', 'Built_2005_or_later'))
#acsub

cor(acsub, use = "complete.obs", method = "pearson")
```

```
##                      Median_house_value Built_2005_or_later
## Median_house_value          1.00000000         -0.03690554
## Built_2005_or_later        -0.03690554          1.00000000
```

```
#santa clara county

scsub = subset(ca_pa, COUNTYFP == 85, select = c('Median_house_value', 'Built_2005_or_later'))
#scsub

cor(scsub, use = "complete.obs", method = "pearson")
```

```
##                      Median_house_value Built_2005_or_later
## Median_house_value          1.00000000         -0.08501218
## Built_2005_or_later        -0.08501218          1.00000000
```

```
#Allegheny County

alsub = subset(ca_pa, COUNTYFP == 3, select = c('Median_house_value', 'Built_2005_or_later'))
#alsub

cor(alsub, use = "complete.obs", method = "pearson")
```

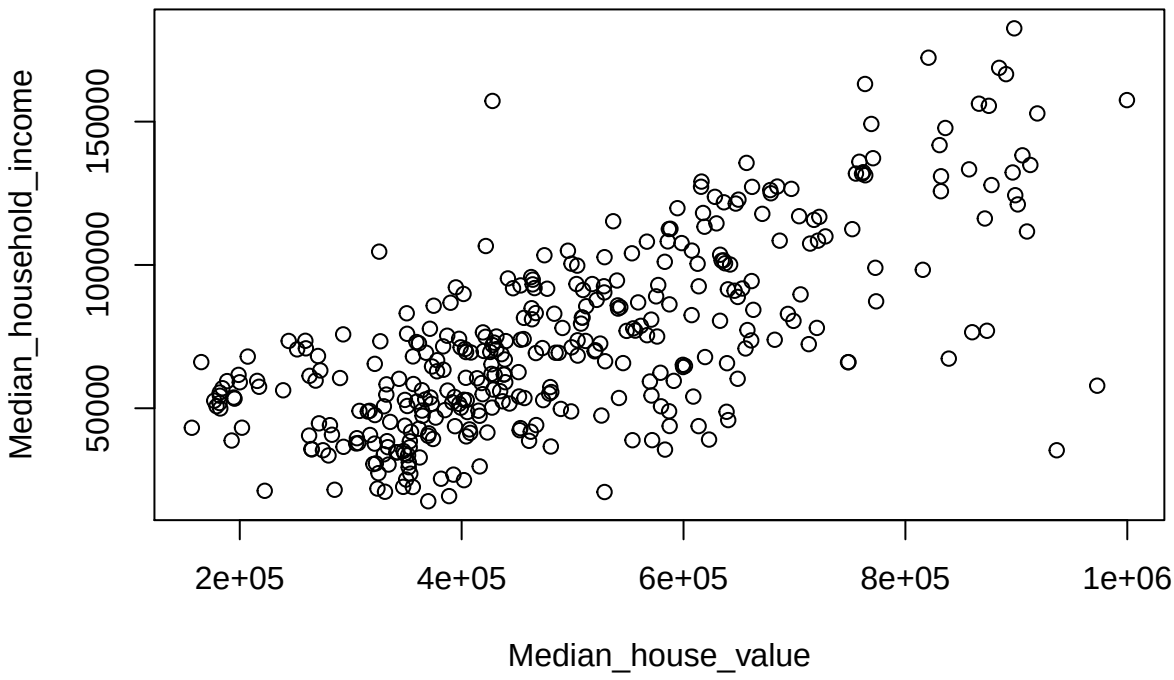
```
##                      Median_house_value Built_2005_or_later
## Median_house_value           1.0000000           0.1925676
## Built_2005_or_later          0.1925676           1.0000000
```

e. Make three plots, showing median house values against median income, for Alameda, Santa Clara, and Al

```
#I will try to condense the code a little bit here by using the $ to call out the columns I want after
# making a subset of the data for the desired counties

# alameda county

alcounty = subset(ca_pa, COUNTYFP == 1, select = c('Median_house_value', 'Median_household_income'))
plot(alcounty)
```

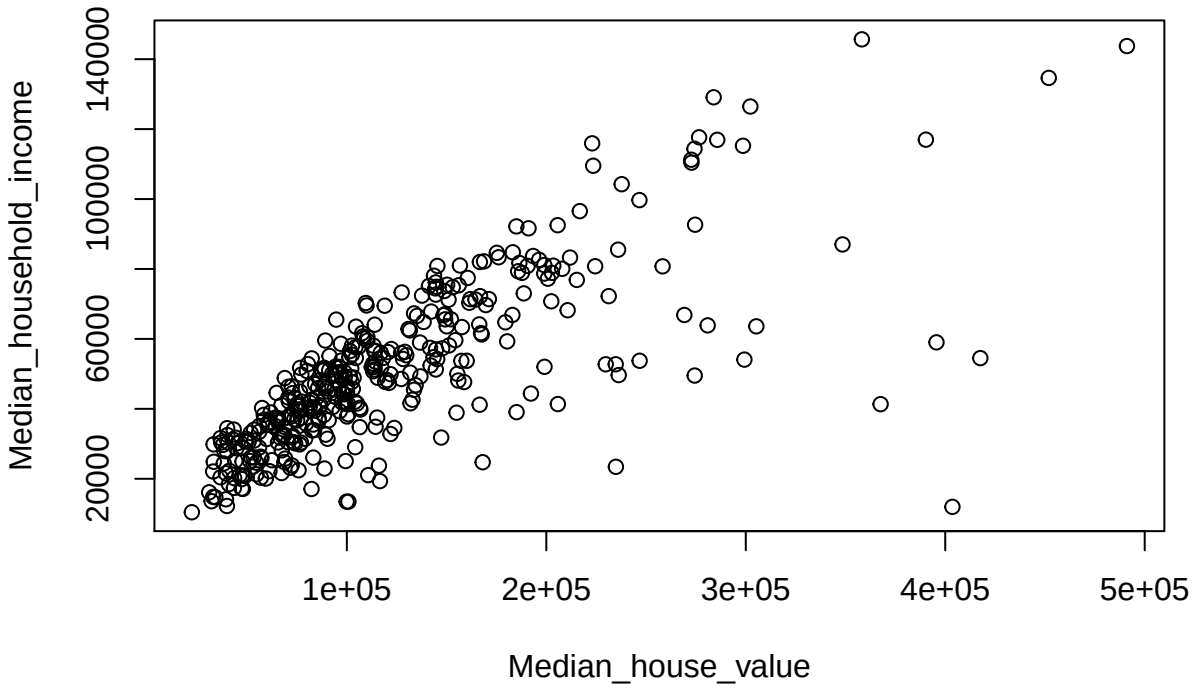



```
# santa clara county

sccounty = subset(ca_pa, COUNTYFP == 85, select = c('Median_house_value', 'Median_household_income'))
plot(sccounty)
```


```
# allegheny county

allecounty = subset(ca_pa, COUNTYFP == 3, select = c('Median_house_value', 'Median_household_income'))
plot(allecounty)
```




```
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])
}
acca
```