Homework 2: Housing Price

DingHuan, 3170102085

1. *Loading and cleaning*

   a.

```r
ca_pa <- read.csv("data/calif_penn_2011.csv",header = T)
```

   b. The dataframe has 11275 rows and 34 columns.

```r
dim(ca_pa)
```

```
## [1] 11275    34
```

   c. The results below are hidden given that it is too long to show them all. `apply(ca_pa,c(1,2),is.na)` returns a matrix having the same dimension as `ca_pa`, whose elements are Boolean numbers indicating whether the data in `ca_pa` is NA. `colSums()` sums the columns and returns a named vector indicating how many NA elements are there in each column of `ca_pa`.

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

   d.

```r
ca_pa <- na.omit(ca_pa)
```

   e. There are 670 rows containing NA elements, which is now removed.

```r
11275 - nrow(ca_pa)
```

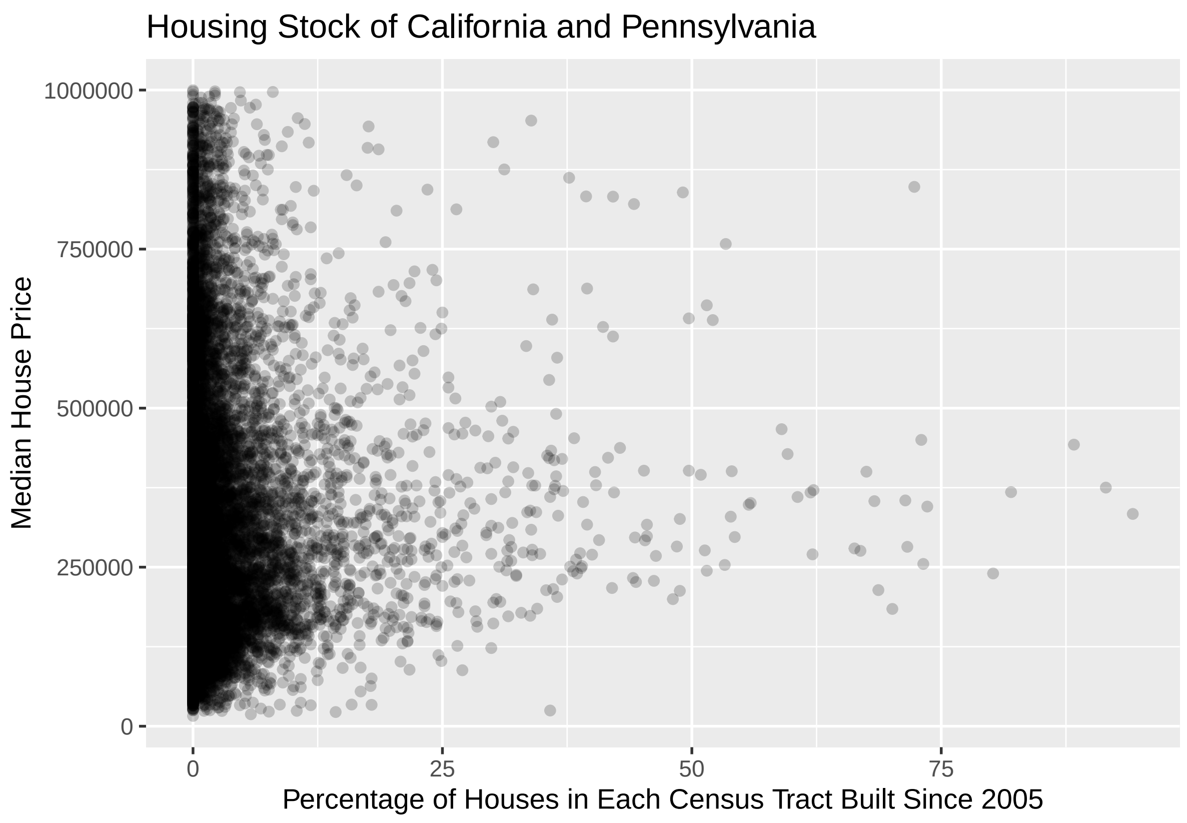
```
## [1] 670
```

   f. The answers in (c) and (e) are compatible. Although we know the number of NA elements in each column, we still have no idea whether they are in the same row or not.

2. *This Very New House*

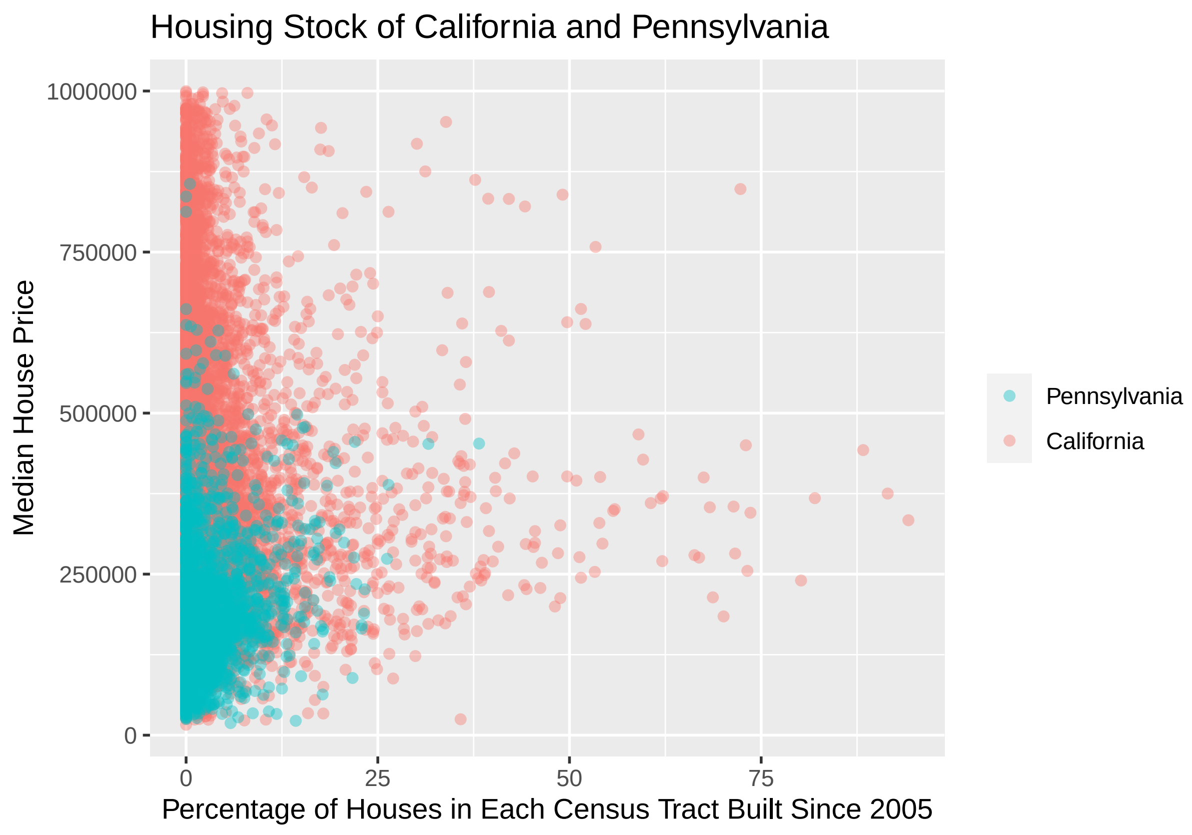
   a.

```r
library(tidyverse)
ca_pa %>% ggplot(aes(x = Built_2005_or_later, y = Median_house_value)) +
  geom_point(alpha = 0.2) +
  labs(x = "Percentage of Houses in Each Census Tract Built Since 2005",
       y = "Median House Price",
       title = "Housing Stock of California and Pennsylvania")
```



b.

```
ca_pa %>% ggplot() +
  geom_point(aes(x = Built_2005_or_later, y = Median_house_value,
                 color = (STATEFP==42)),
             alpha = 0.4) +
  labs(x = "Percentage of Houses in Each Census Tract Built Since 2005",
       y = "Median House Price",
       title = "Housing Stock of California and Pennsylvania") +
  scale_colour_hue(element_blank(), breaks=c(TRUE, FALSE),
                   labels=c("Pennsylvania","California"))
```



3. *Nobody Home*

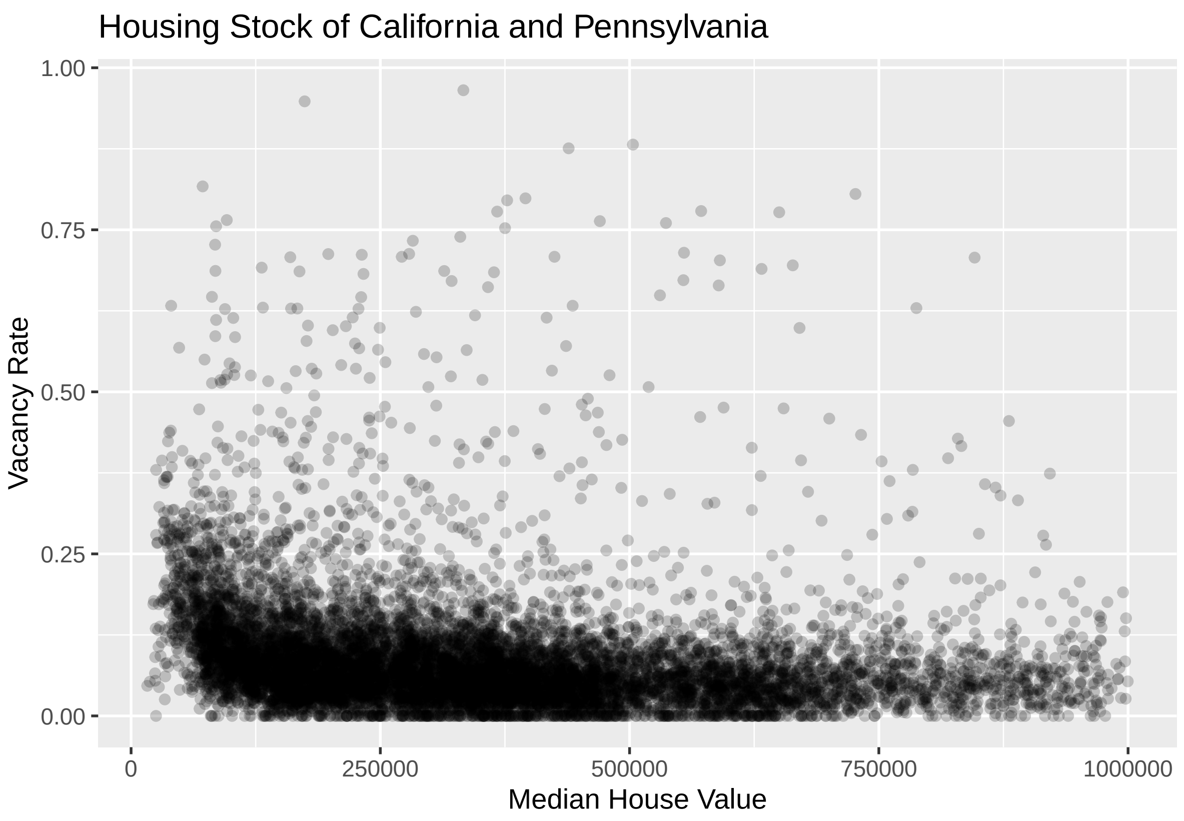
   a. The minimum the minimum,maximum, mean, and median vacancy rates are 0.00000, 0.96531, 0.08889 and 0.06767 respectively.

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

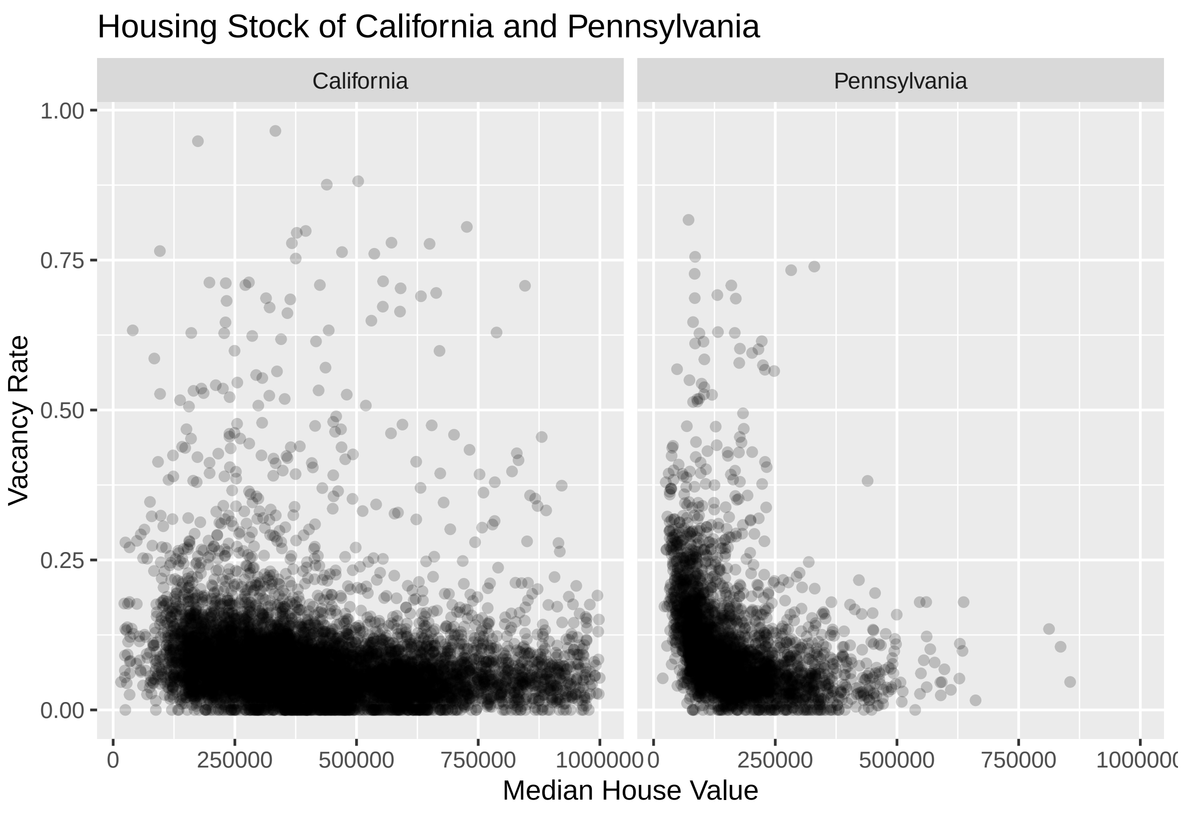
   b.

```
ca_pa %>% ggplot(aes(x = Median_house_value, y = Vacancy_rate)) +
  geom_point(alpha = 0.2) +
  labs(x = "Median House Value",
       y = "Vacancy Rate",
       title = "Housing Stock of California and Pennsylvania")
```



c. It is clear that there are fewer Census tracts with high median house value in Pennsylvania, among which there are more Census tracts have higher vacancy rate. Although the number of Census tracts with high median house value in Pennsylvania is not high, their vacancy rate is much lower than those in California. The distribution of vacancy rate among Census tracts in California seems not to change with median house value.

```
ca_pa %>% ggplot(aes(x = Median_house_value, y = Vacancy_rate)) +
  geom_point(alpha = 0.2) +
  labs(x = "Median House Value",
       y = "Vacancy Rate",
       title = "Housing Stock of California and Pennsylvania") +
  facet_wrap(~ STATEFP,
             labeller = as_labeller(c('6' = "California", '42' = "Pennsylvania")))
```



4. a. The first iteration records the row numbers of the county marked as 1 in California to the variable `acca`. The second iteration records the median house value of the Census tracts in `acca` to the variable `accamhv`, and finally calculate thier median value (the median value of the median values of the Census tracts recorded in `accamhv`).

```r
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)
```

```
## [1] 474050
```

b.

```r
median(ca_pa[ca_pa$STATEFP == 6 & ca_pa$COUNTYFP == 1, "Median_house_value"])
```

```
## [1] 474050
```

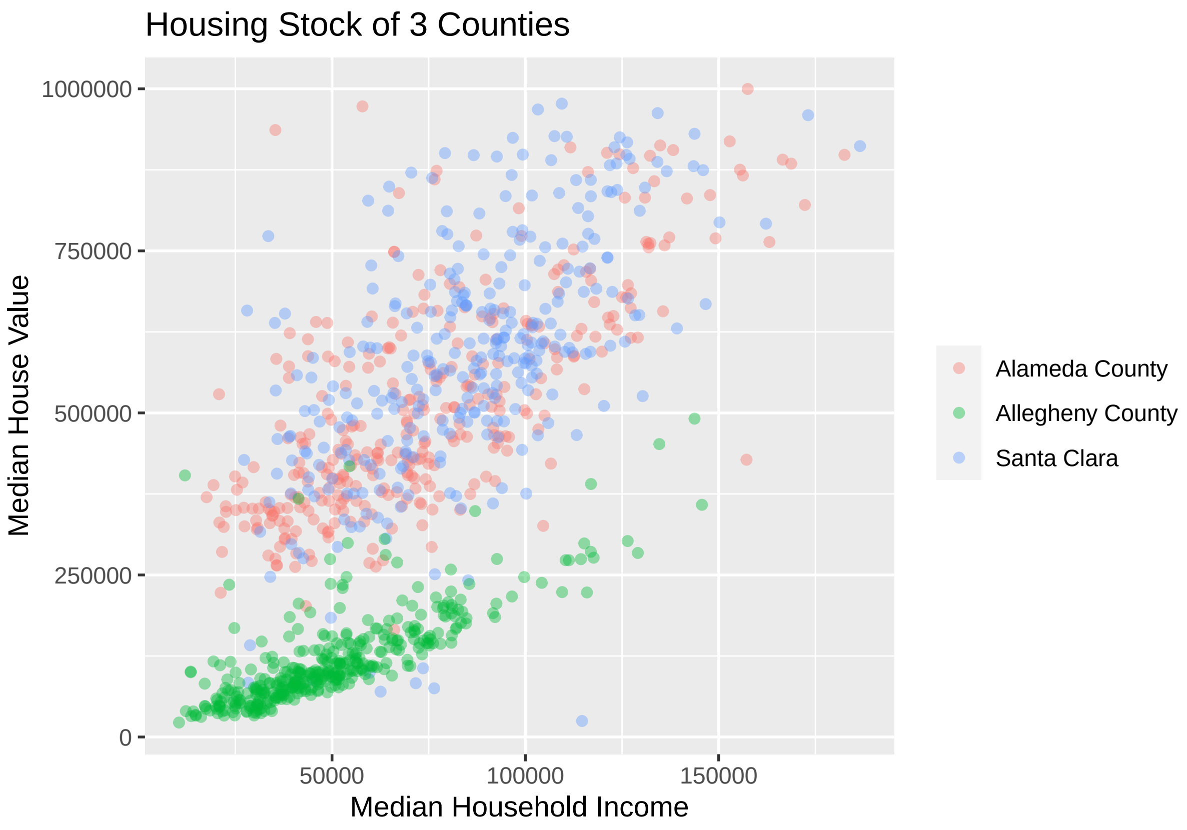
c.

```r
ca_pa_long <- ca_pa %>%
  gather(key = built_year,
         value = built_numbers,
         dplyr::starts_with('Built'))
```

d.

e.

```
ca_pa %>% filter(((COUNTYFP==1|COUNTYFP==85)&STATEFP == 6) | COUNTYFP==3&STATEFP == 42) %>%
  mutate(COUNTYNAME = ifelse(COUNTYFP == 1, "Alameda County",
                             ifelse(COUNTYFP == 85, "Santa Clara", "Allegheny County"))) %>%
  ggplot(aes(x = Median_household_income, y=Median_house_value, color = COUNTYNAME)) +
  geom_point(alpha = 0.4) +
  labs(x = "Median Household Income",
       y = "Median House Value",
       title = "Housing Stock of 3 Counties") +
  theme(legend.title=element_blank())
```



5. (MB.CH1.11) The first line create an variable `gender` with 2 levels "female" and "male", whose first 91 elements are "female" and the remaining elements are "male". The second line `table(gender)` shows the factor levels. The third line exchanges the order of levels. The function searches `gender` first, finding the same level as elements in `levels=c("male", "female")`, and then changes the numeric order to the new levels. But when it doesn't find the same level, just as line 5, the function removes the old levels and create an new level named `Male` and matches nothing, thus the result of `table(gender)` is 0 for level "Male". When NA is included in table, like what line8 does, we can see an NA level with 92 elements, which are exactly those whose level "male" are removed in line5.

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

```
## gender
## female   male
##     91     92
```

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

```
## gender
##   male female
##     92     91
```

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

```
## gender
##   Male female
##      0     91
```

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

```
## gender
##   Male female   <NA>
##      0     91     92
```

```
rm(gender)  # Remove gender
```

6. (MB.CH.1.12)

   a.

```
cutoff <- function(x, value){
prop = sum(x > value) / length(x)
return(prop)
}
cutoff(1:100, 10)
```

```
## [1] 0.9
```

```
cutoff(1:100, 35)
```

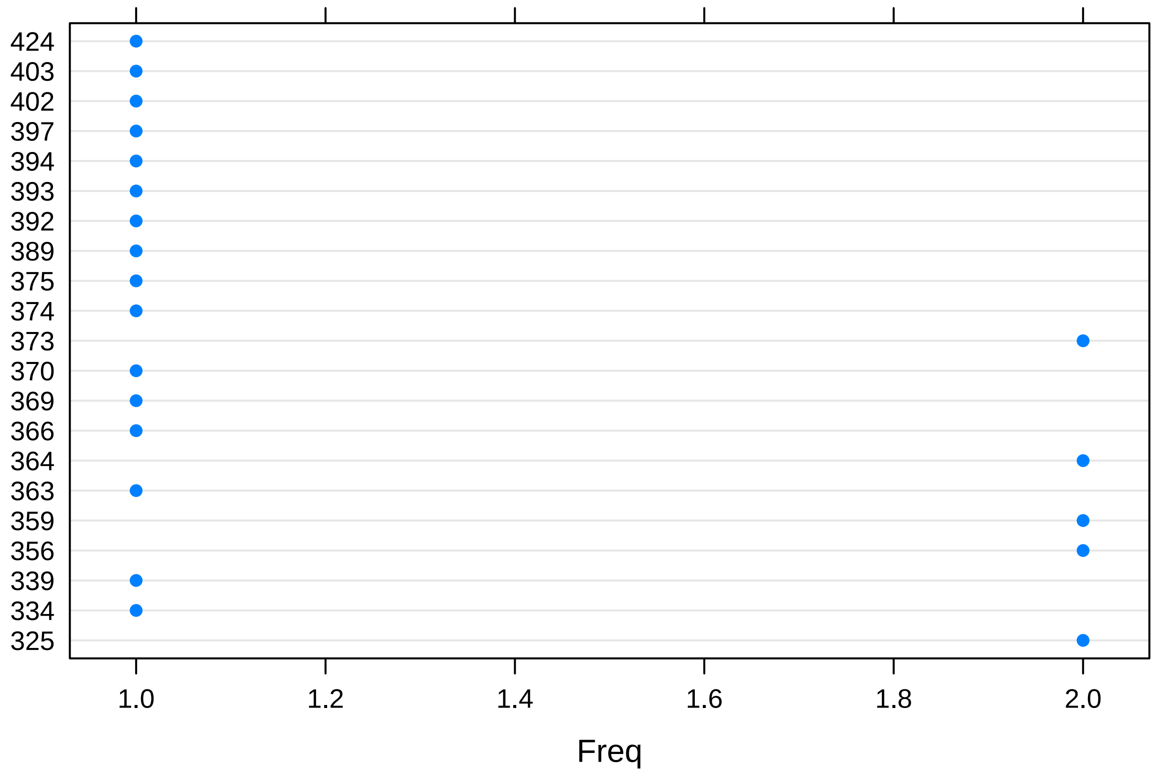
```
## [1] 0.65
```

```
cutoff(1:100, 35.5)
```

```
## [1] 0.65
```

   b.

```
library(Devore7)
dotplot(ex01.36)
```



```r
cutoff(ex01.36$C1, 420)
```

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