Ch2: Statistical Learning Applied Exercises

Kylie

30/03/2019

Exercise 8

This exercise relates to the `College` data set, which can be found in the file `College.csv`. It contains a number of variables for 777 different universities and colleges in the US. The variables are - `Private`: Public/private indicator - `Apps`: Number of applications received - `Accept`: Number of applicants accepted - `Enroll`: Number of new students enrolled - `Top10perc`: New students from top 10% of high school class - `Top25perc`: New students from top 25% of high school class - `F.Undergrad`: Number of full-time undergraduates - `P.Undergrad`: Number of part-time undergraduates - `Outstate`: Out-of-state tuition - `Room.Board`: Room and board costs - `Books`: Estimated book costs - `Personal`: Estimated personal spending - `PhD`: Percent of faculty with Ph.D.'s - `Terminal`: Percent of faculty with terminal degree - `S.F.Ratio`: Student/faculty ratio - `perc.alumni`: Percent of alumni who donate - `Expend`: Instructional expenditure per student - `Grad.Rate`: Graduation rate

(a) Use the `read.csv()` function to read the data into R. Call the loaded data college. Make sure that you have the directory set to the correct location for the data.

```
# suppressing messages from readr
options(readr.num_columns = 0)

college <- read_csv('College.csv')
```

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

```
sum(is.na(college)) # checking for missing values (this should be zero)
```

```
## [1] 0
```

(b) Look at the data using the `fix()` function. You should notice that the first column is just the name of each university.We don't really want R to treat this as data. However, it may be handy to have these names for later. Try the following commands:

```
rownames (college )=college [,1] fix (college )
```

You should see that there is now a row.names column with the name of each university recorded. This means that R has given each row a name corresponding to the appropriate university. R will not try to perform calculations on the row names. However, we still need to eliminate the first column in the data where the names are stored. Try `college =college [,-1] fix (college )`

Now you should see that the first data column is Private. Note that another column labeled row.names now appears before the Private column. However, this is not a data column but rather the name that R is giving to each row.

```
# suppressing messages from readr
options(readr.num_columns = 0)

college <- read_csv('College.csv')
```

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

```
sum(is.na(college)) # checking for missing values (this should be zero)
```

```
## [1] 0
```

```
#fix(college)
college <- as.data.frame(college)
rownames(college) <- college[,1]
#fix (college)

college <- college[,-1]
#fix(college)
```

(c) i. Use the `summary()` function to produce a numerical summary of the variables in the data set.

```
summary(college)
```

```
##    Private                Apps           Accept          Enroll
##  Length:777         Min.   :   81   Min.   :   72   Min.   :  35
##  Class :character   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242
##  Mode  :character   Median : 1558   Median : 1110   Median : 434
##                     Mean   : 3002   Mean   : 2019   Mean   : 780
##                     3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902
##                     Max.   :48094   Max.   :26330   Max.   :6392
##    Top10perc       Top25perc      F.Undergrad     P.Undergrad
##  Min.   : 1.00   Min.   :  9.0   Min.   :  139   Min.   :    1.0
##  1st Qu.:15.00   1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0
##  Median :23.00   Median : 54.0   Median : 1707   Median :  353.0
##  Mean   :27.56   Mean   : 55.8   Mean   : 3700   Mean   :  855.3
##  3rd Qu.:35.00   3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0
##  Max.   :96.00   Max.   :100.0   Max.   :31643   Max.   :21836.0
##     Outstate       Room.Board       Books           Personal
##  Min.   : 2340   Min.   :1780   Min.   :  96.0   Min.   : 250
##  1st Qu.: 7320   1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850
##  Median : 9990   Median :4200   Median : 500.0   Median :1200
##  Mean   :10441   Mean   :4358   Mean   : 549.4   Mean   :1341
##  3rd Qu.:12925   3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700
##  Max.   :21700   Max.   :8124   Max.   :2340.0   Max.   :6800
##       PhD            Terminal       S.F.Ratio      perc.alumni
##  Min.   :  8.00   Min.   : 24.0   Min.   : 2.50   Min.   : 0.00
##  1st Qu.: 62.00   1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00
##  Median : 75.00   Median : 82.0   Median :13.60   Median :21.00
##  Mean   : 72.66   Mean   : 79.7   Mean   :14.09   Mean   :22.74
##  3rd Qu.: 85.00   3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00
##  Max.   :103.00   Max.   :100.0   Max.   :39.80   Max.   :64.00
##      Expend        Grad.Rate
##  Min.   : 3186   Min.   : 10.00
##  1st Qu.: 6751   1st Qu.: 53.00
##  Median : 8377   Median : 65.00
##  Mean   : 9660   Mean   : 65.46
##  3rd Qu.:10830   3rd Qu.: 78.00
##  Max.   :56233   Max.   :118.00
```

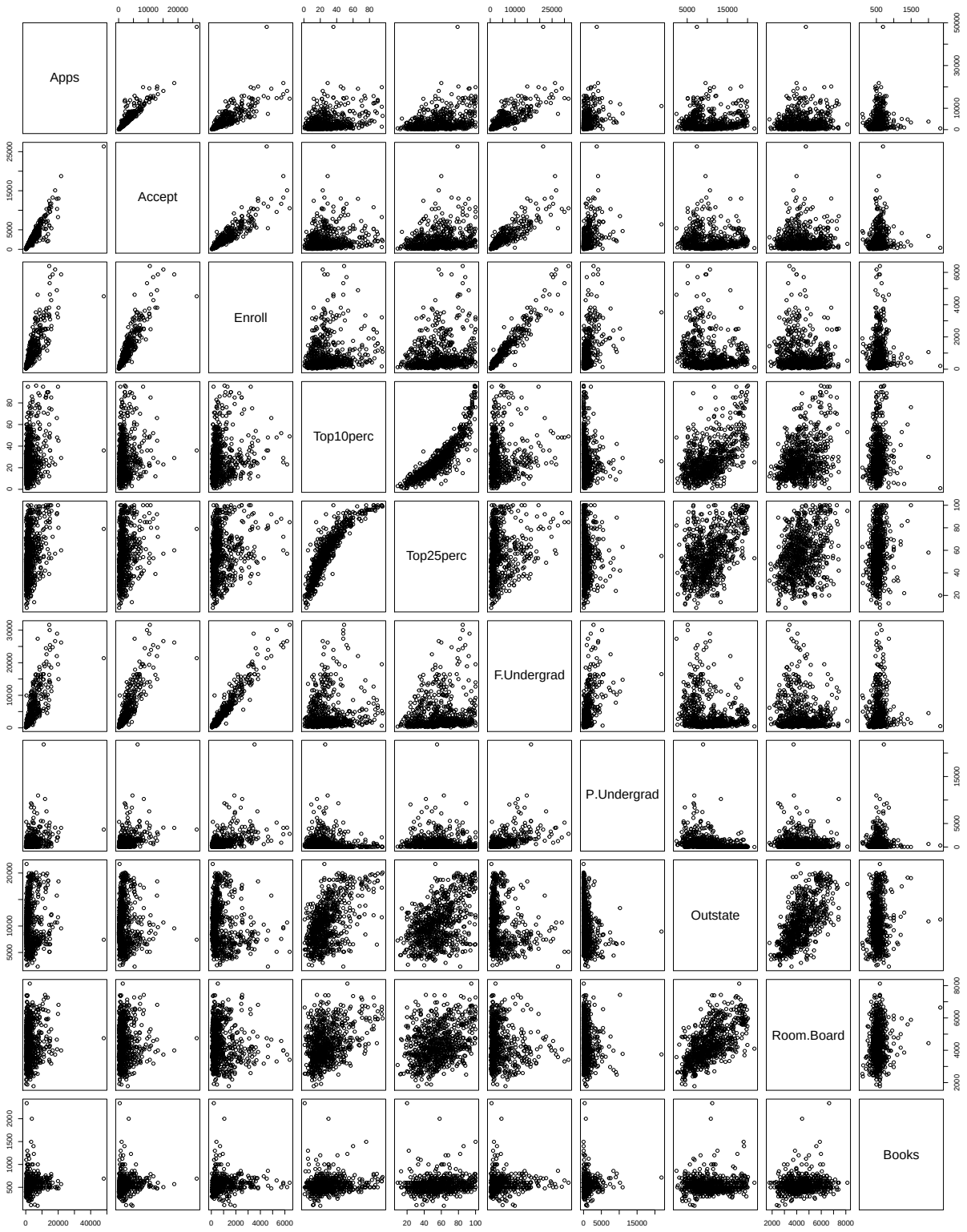
```
skim(college)
```

```
## Skim summary statistics
##  n obs: 777
##  n variables: 18
##
```

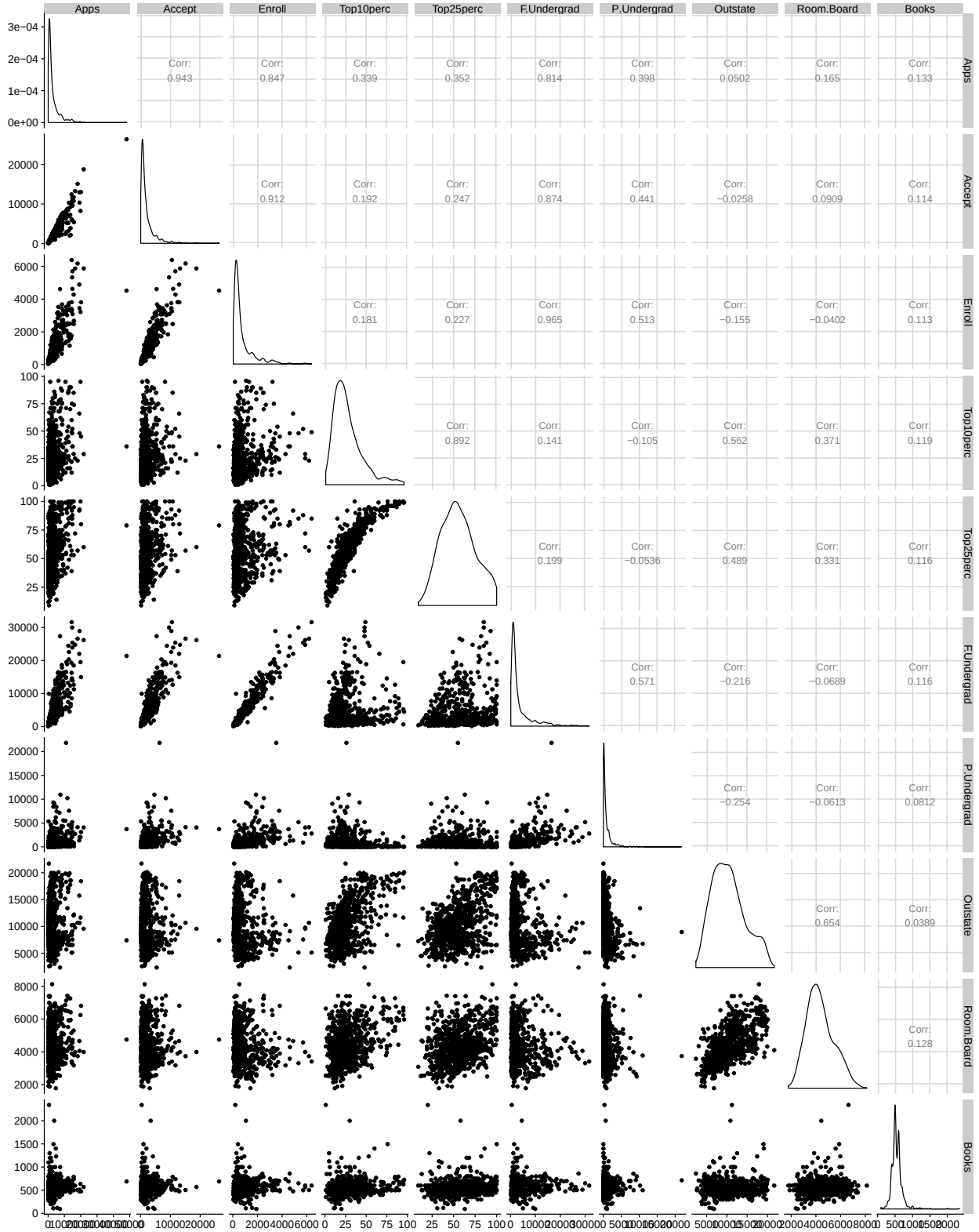
```
## -- Variable type:character -----------------------------------------------------------------
##  variable missing complete   n min max empty n_unique
##   Private       0      777 777   2   3     0        2
##
## -- Variable type:numeric -------------------------------------------------------------------
##     variable missing complete   n     mean      sd     p0    p25    p50
##       Accept       0      777 777  2018.8  2451.11   72    604   1110
##         Apps       0      777 777  3001.64 3870.2    81    776   1558
##        Books       0      777 777   549.38  165.11   96    470    500
##       Enroll       0      777 777   779.97  929.18   35    242    434
##       Expend       0      777 777  9660.17 5221.77 3186   6751   8377
##  F.Undergrad       0      777 777  3699.91 4850.42  139    992   1707
##    Grad.Rate       0      777 777    65.46   17.18   10     53     65
##     Outstate       0      777 777 10440.67 4023.02 2340   7320   9990
##  P.Undergrad       0      777 777   855.3  1522.43    1     95    353
##  perc.alumni       0      777 777    22.74   12.39    0     13     21
##     Personal       0      777 777  1340.64  677.07  250    850   1200
##          PhD       0      777 777    72.66   16.33    8     62     75
##   Room.Board       0      777 777  4357.53 1096.7  1780   3597   4200
##    S.F.Ratio       0      777 777    14.09    3.96    2.5   11.5   13.6
##     Terminal       0      777 777    79.7    14.72   24     71     82
##    Top10perc       0      777 777    27.56   17.64    1     15     23
##    Top25perc       0      777 777    55.8    19.8     9     41     54
##      p75    p100     hist
##   2424   26330   <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##   3624   48094   <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##    600    2340   <U+2581><U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##    902    6392   <U+2587><U+2582><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##  10830   56233   <U+2587><U+2583><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##   4005   31643   <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##     78     118   <U+2581><U+2581><U+2583><U+2587><U+2587><U+2585><U+2582><U+2581>
##  12925   21700   <U+2582><U+2586><U+2587><U+2587><U+2585><U+2582><U+2582><U+2581>
##    967   21836   <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##     31      64   <U+2583><U+2587><U+2587><U+2586><U+2583><U+2582><U+2581><U+2581>
##   1700    6800   <U+2587><U+2587><U+2583><U+2581><U+2581><U+2581><U+2581><U+2581>
##     85     103   <U+2581><U+2581><U+2581><U+2583><U+2586><U+2587><U+2587><U+2583>
##   5050    8124   <U+2581><U+2585><U+2587><U+2587><U+2583><U+2582><U+2581><U+2581>
##     16.5    39.8 <U+2581><U+2585><U+2587><U+2583><U+2581><U+2581><U+2581><U+2581>
##     92     100   <U+2581><U+2581><U+2581><U+2582><U+2583><U+2586><U+2586><U+2587>
##     35      96   <U+2583><U+2587><U+2586><U+2582><U+2582><U+2581><U+2581><U+2581>
##     69     100   <U+2581><U+2583><U+2586><U+2587><U+2587><U+2585><U+2583><U+2582>
```

ii. Use the `pairs()` function to produce a scatterplot matrix of the first ten columns or variables of the data. Recall that you can reference the first ten columns of a matrix `A` using `A[,1:10]`.

```
pairs(select(college, 2:11))
```

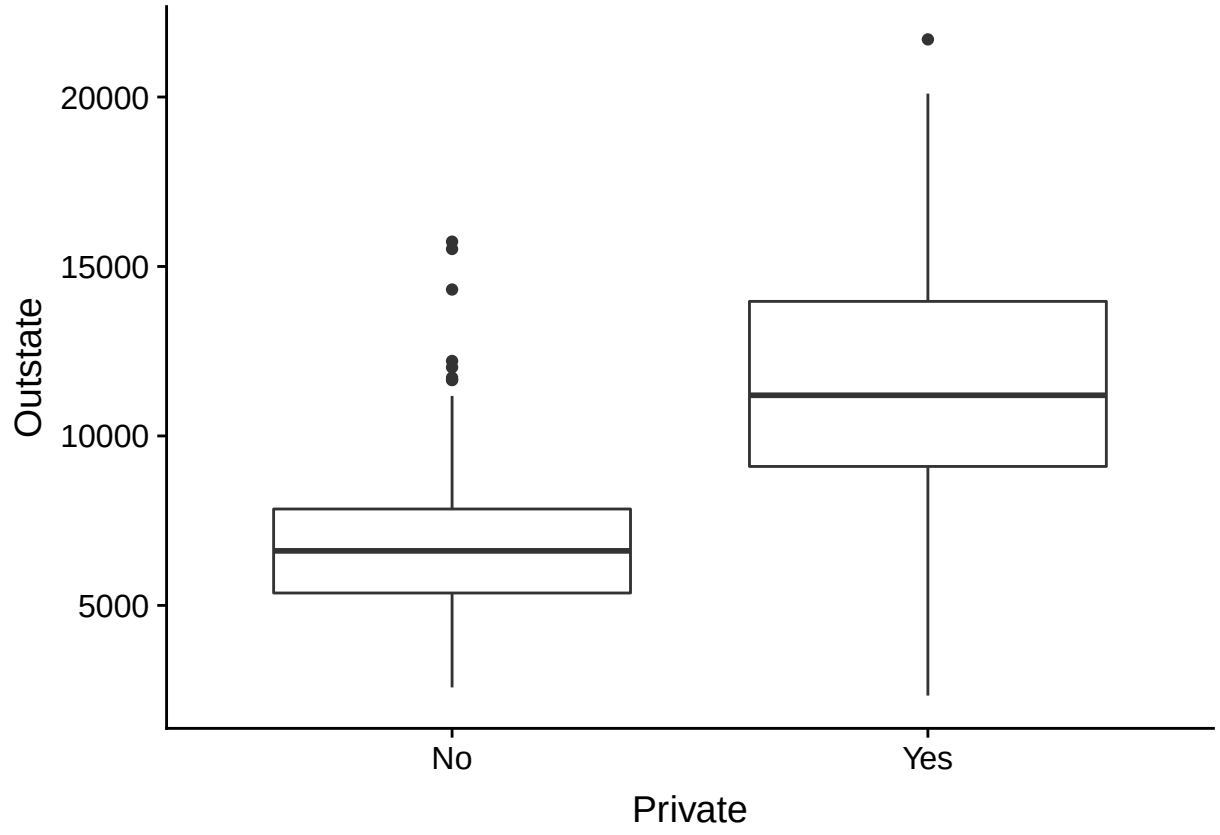



```
# Plotting matrix of pair-wise scatterplots of first 10 variables
ggpairs(select(college, 2:11))
```



iii. Use the `plot()` function to produce side-by-side boxplots of Outstate versus Private.

```
ggplot(college, mapping = aes(x =Private, y = Outstate)) +
  geom_boxplot()
```



iv. Create a new qualitative variable, called `Elite`, by binning the `Top10perc` variable. We are going to divide universities into two groups based on whether or not the proportion of students coming from the top 10% of their high school classes exceeds 50%.

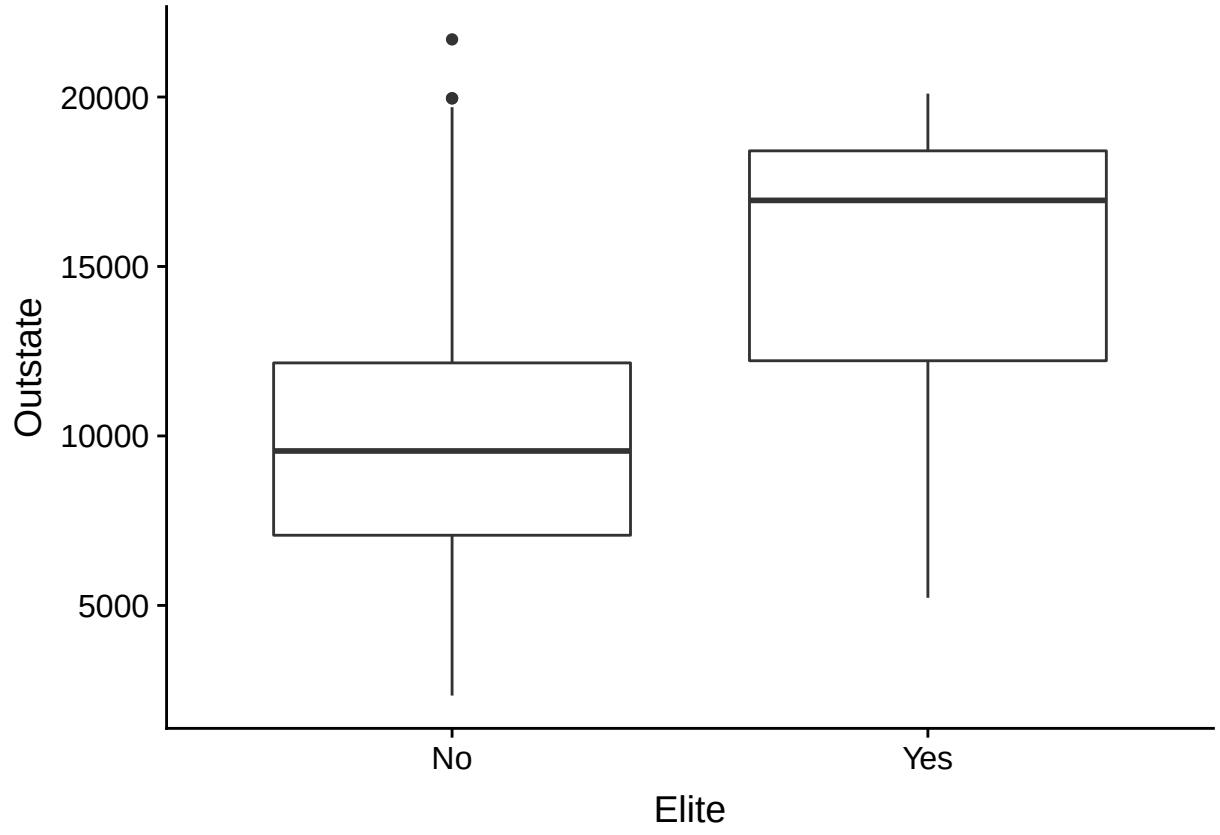
```
Elite =rep ("No",nrow(college ))

Elite [college$Top10perc >50]=" Yes"

Elite =as.factor (Elite)

college =data.frame(college ,Elite)
```

Use the `summary()` function to see how many elite universities there are. Now use the `plot()` function to produce side-by-side boxplots of `Outstate` versus `Elite`.

```
college <- mutate(college, Elite = as.factor(case_when(Top10perc > 50 ~ "Yes",
                            Top10perc <= 50 ~ "No")))

skim(select(college, Elite))
```

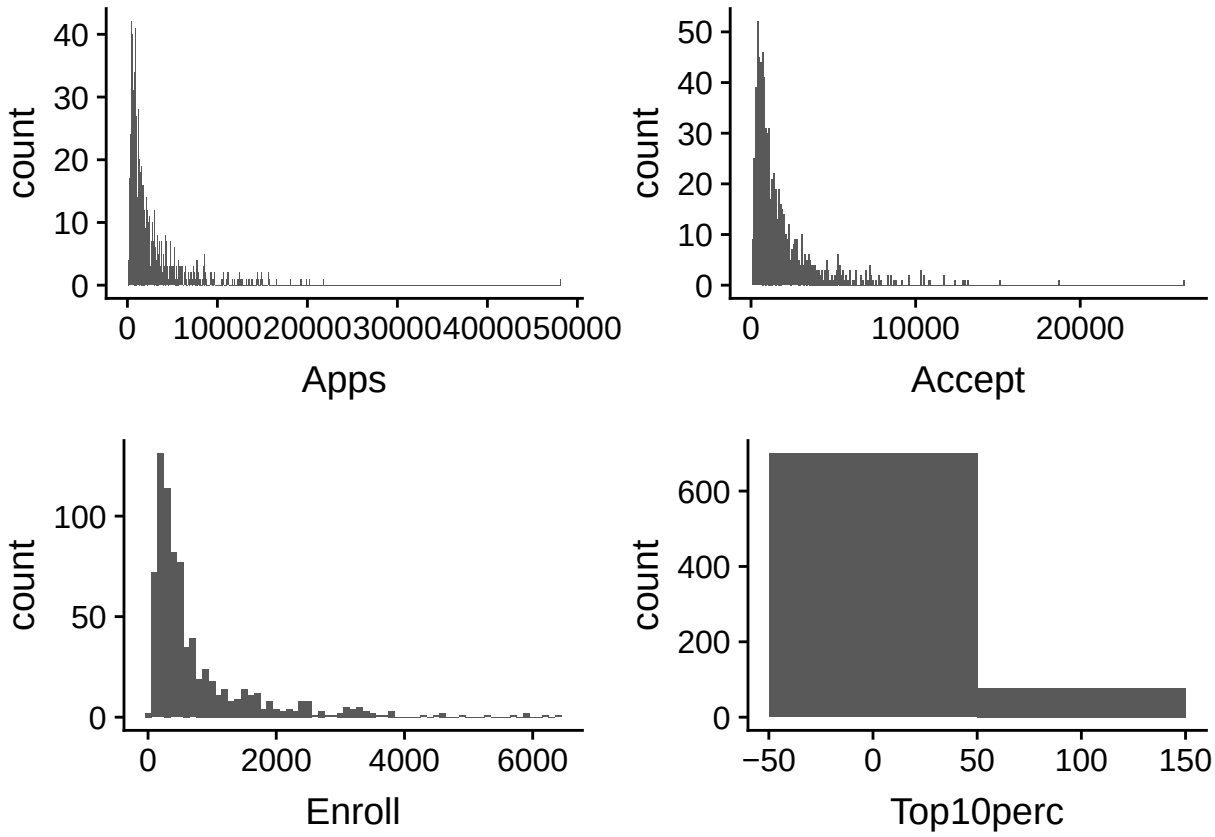
```
## Skim summary statistics
##  n obs: 777
##  n variables: 1
##
## -- Variable type:factor ------------------------------------------------------------------------
##  variable missing complete   n n_unique              top_counts ordered
##     Elite       0      777 777        2 No: 699, Yes: 78, NA: 0   FALSE
```

```
ggplot(college, mapping = aes(x = Elite, y = Outstate)) +
  geom_boxplot()
```

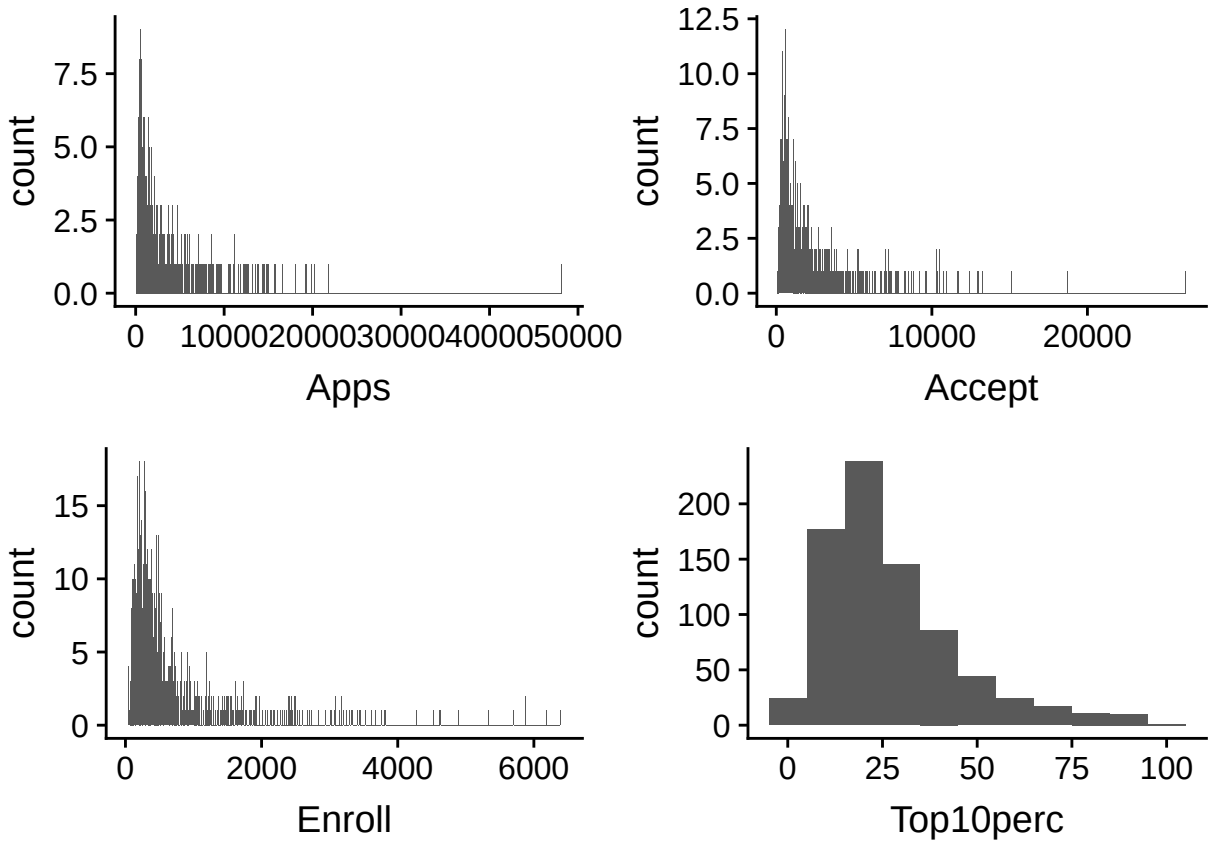


v. Use the `hist()` function to produce some histograms with differing numbers of bins for a few of the quantitative variables. You may find the command `par(mfrow=c(2,2))` useful: it will divide the print window into four regions so that four plots can be made simultaneously. Modifying the arguments to this function will divide the screen in other ways.

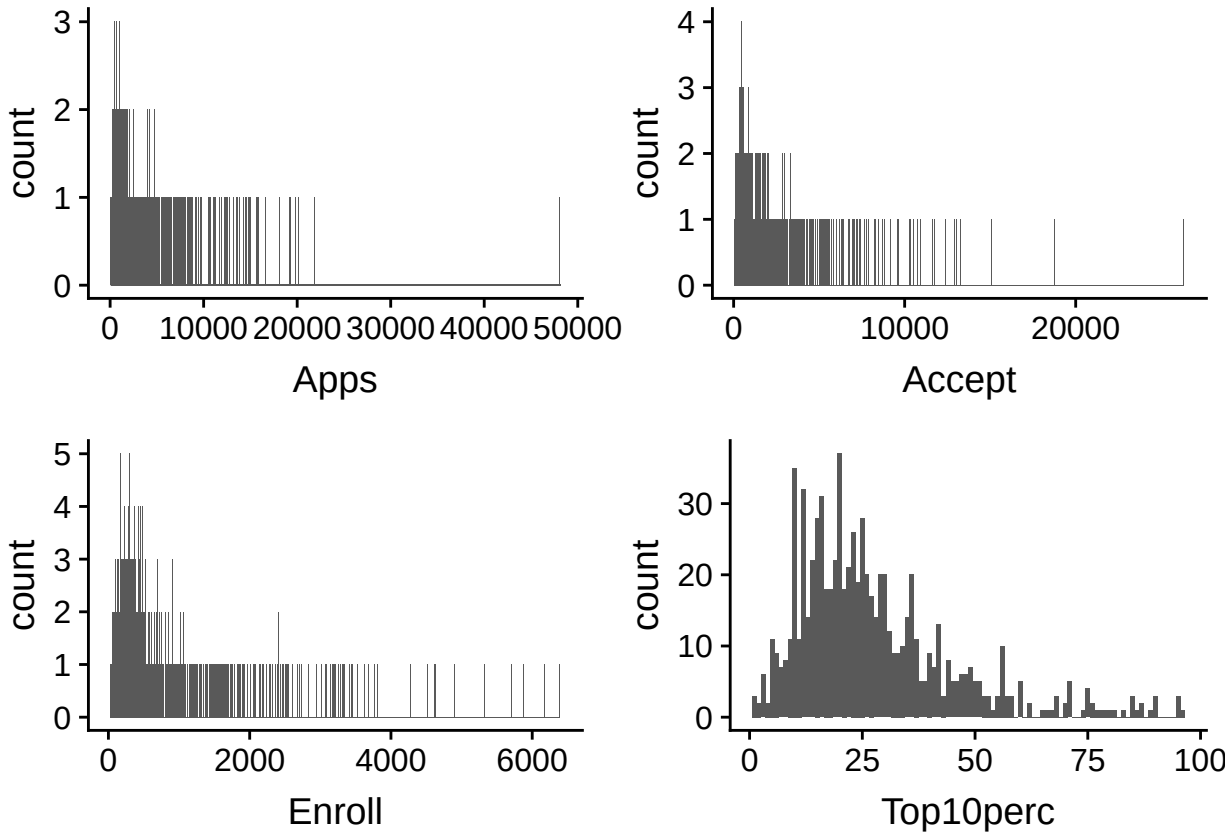
```
# trying bin width of 100
A <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Apps), binwidth = 100)

B <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Accept), binwidth = 100)

C <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Enroll), binwidth = 100)

D <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Top10perc), binwidth = 100)

plot_grid(A, B, C, D)
```



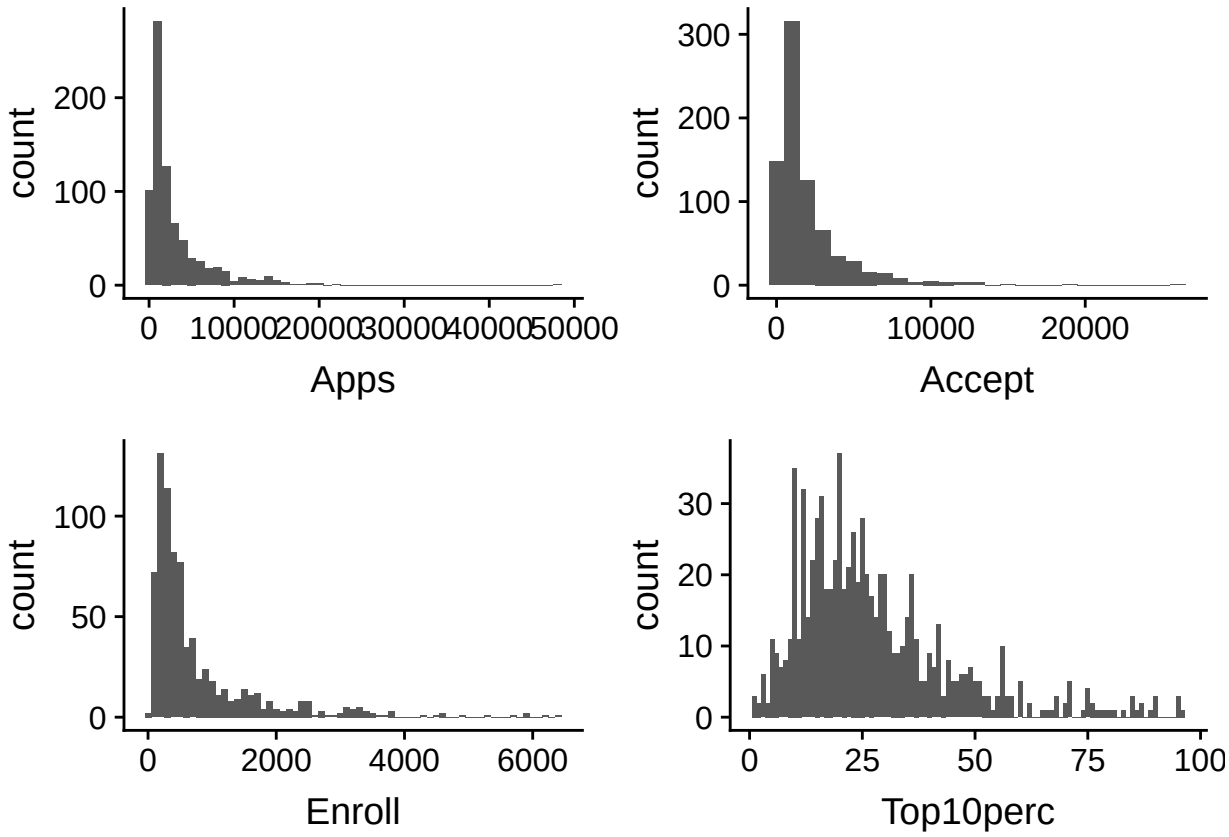

```
# trying bin width of 10
A <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Apps), binwidth = 10)

B <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Accept), binwidth = 10)

C <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Enroll), binwidth = 10)

D <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Top10perc), binwidth = 10)

plot_grid(A, B, C, D)
```




```
# trying bin width of 1
A <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Apps), binwidth = 1)

B <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Accept), binwidth = 1)

C <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Enroll), binwidth = 1)

D <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Top10perc), binwidth = 1)

plot_grid(A, B, C, D)
```




```
# trying a mix of bin widths
A <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Apps), binwidth = 1000)

B <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Accept), binwidth = 1000)

C <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Enroll), binwidth = 100)

D <- ggplot(data = college) +
  geom_histogram(mapping = aes(x = Top10perc), binwidth = 1)

plot_grid(A, B, C, D)
```



vi. Continue exploring the data, and provide a brief summary of what you discover.

Exercise 9

This exercise involves the `Auto` data set studied in the lab. Make sure that the missing values have been removed from the data.

(a) Which of the predictors are quantitative, and which are qualitative?

```
# suppressing messages from readr
options(readr.num_columns = 0)

Auto <- read_csv("Auto.csv", na = c("", "NA", "?")) # loading data, making sure missing values (?s) are
Auto <- na.omit(Auto) # removing missing values
sum(is.na(Auto)) # checking for missing values (this should be zero)
```

```
## [1] 0
```

```
Auto <- mutate_if(Auto, is.character, factor) # converting characters to factors
Auto <- mutate(Auto, origin = as.factor(origin)) #converting origin to factor
```

Use `View(Auto)` to look at all of the data.

`mpg`, `displacement`, `horsepower`, `weight`, `acceleration`, and `year` are quantitative variables. `origin` and `name` are qualitative. `cylinder` appears to be quantiative, but only takes a few vaues (8, 4, 6, 3, 5), so it may be better to treat this as a qualitative variable

(b) What is the range of each quantitative predictor? You can answer this using the `range()` function.

Including `cylinder` as quantitative for now.

```
max_Auto <- summarise_if(Auto, is.numeric, list(max)) # calculating max for all numeric variables in Au

min_Auto <- summarise_if(Auto, is.numeric, list(min)) # calculating min for all numeric variables in Au

(range_Auto <- max_Auto - min_Auto) # evaluating and displaying range
```

```
##    mpg cylinders displacement horsepower weight acceleration year
## 1 37.6         5          387        184   3527         16.8   12
```

(c) What is the mean and standard deviation of each quantitative predictor?

```
# Mean:
(mean_Auto <- summarise_if(Auto, is.numeric, list(mean))) # calculating and displaying mean for all num
```

```
## # A tibble: 1 x 7
##     mpg cylinders displacement horsepower weight acceleration  year
##   <dbl>     <dbl>        <dbl>      <dbl>  <dbl>        <dbl> <dbl>
## 1  23.4      5.47         194.       104.  2978.         15.5  76.0
```

```
# Standard deviation:
(sd_Auto <- summarise_if(Auto, is.numeric, list(sd))) # calculating and displaying mean for all numeric
```

```
## # A tibble: 1 x 7
##     mpg cylinders displacement horsepower weight acceleration  year
##   <dbl>     <dbl>        <dbl>      <dbl>  <dbl>        <dbl> <dbl>
## 1  7.81      1.71         105.       38.5   849.         2.76  3.68
```

(d) Now remove the 10th through 85th observations. What is the range, mean, and standard deviation of each predictor in the subset of the data that remains?

```
# Removing 10th to 85th observations:
Auto_short <- slice(Auto, -10:-85) # negatives indicate the rows should be dropped

# Range calculation:
max_Auto <- summarise_if(Auto_short, is.numeric, list(max)) # calculating max for all numeric variables
min_Auto <- summarise_if(Auto_short, is.numeric, list(min)) # calculating min for all numeric variables
(range_Auto <- max_Auto - min_Auto) # evaluating and displaying range
```

```
##    mpg cylinders displacement horsepower weight acceleration year
## 1 35.6         5          387        184   3348         16.3   12
```

```
# Mean:
(mean_Auto <- summarise_if(Auto_short, is.numeric, list(mean))) # calculating and displaying mean for a
```

```
## # A tibble: 1 x 7
##     mpg cylinders displacement horsepower weight acceleration  year
##   <dbl>     <dbl>        <dbl>      <dbl>  <dbl>        <dbl> <dbl>
## 1  24.4      5.37         187.       101.  2936.         15.7  77.1
```

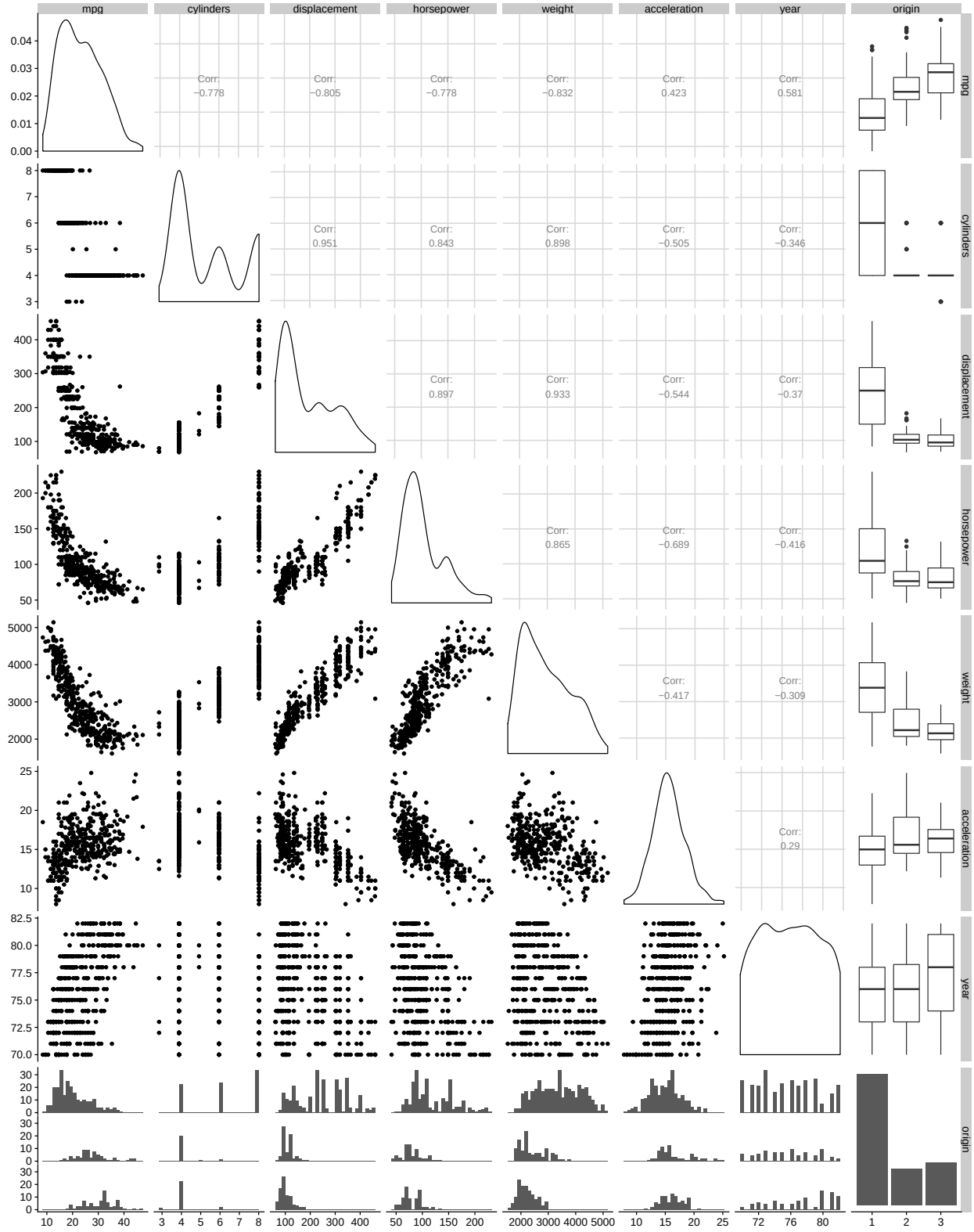
```
# Standard deviation:
(sd_Auto <- summarise_if(Auto_short, is.numeric, list(sd))) # calculating and displaying mean for all n
```

```
## # A tibble: 1 x 7
##     mpg cylinders displacement horsepower weight acceleration  year
##   <dbl>     <dbl>        <dbl>      <dbl>  <dbl>        <dbl> <dbl>
## 1  7.87      1.65         99.7       35.7   811.         2.69  3.11
```

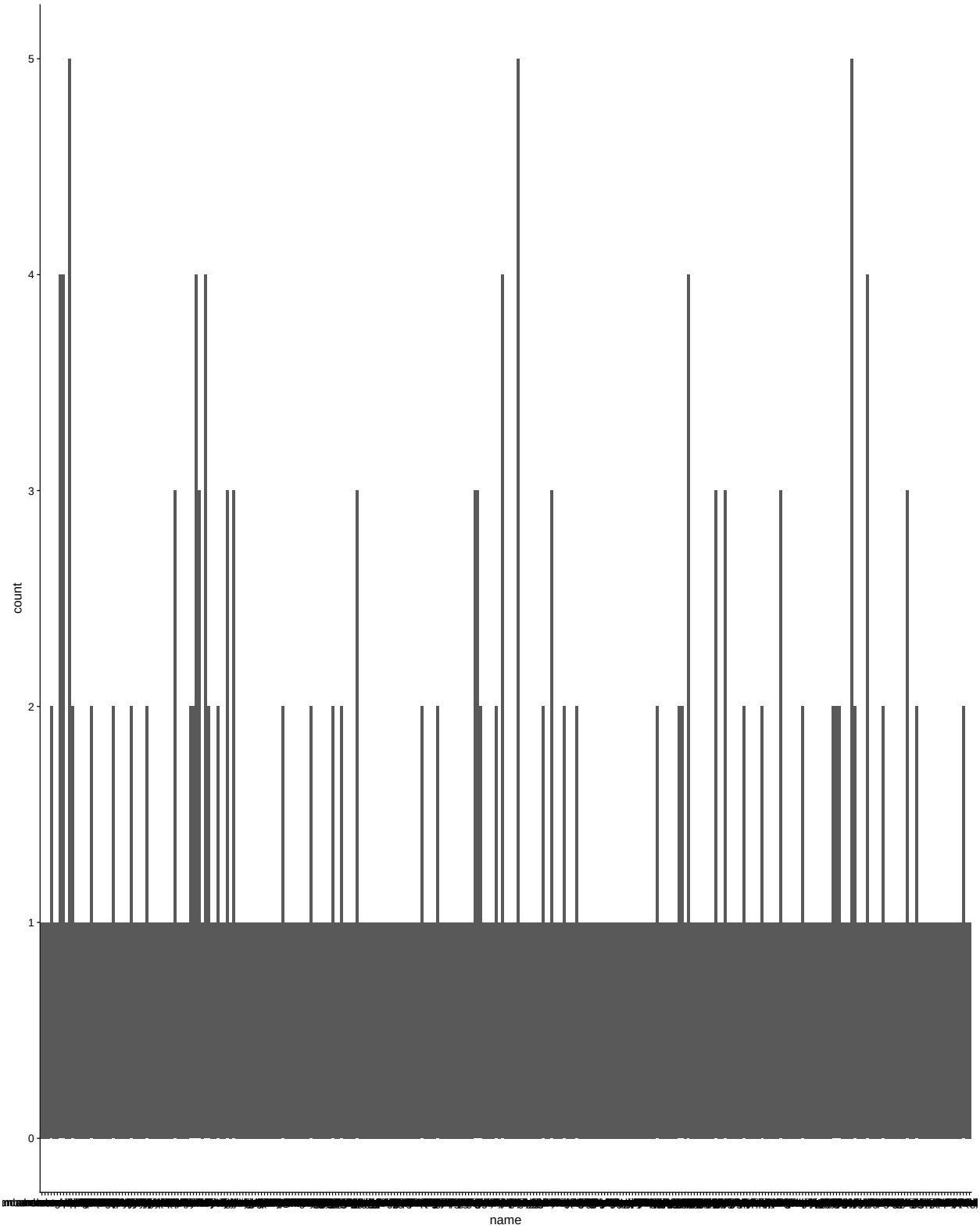
(e) Using the full data set, investigate the predictors graphically, using scatterplots or other

tools of your choice. Create some plots highlighting the relationships among the predictors. Comment on your findings.

```
# suppressing messages from readr
options(readr.num_columns = 0)

Auto <- read_csv("Auto.csv", na = c("", "NA", "?")) # loading data, making sure missing values (?s) are
Auto <- na.omit(Auto) # removing missing values
Auto <- mutate_if(Auto, is.character, factor) # converting characters to factors
Auto <- mutate(Auto, origin = as.factor(origin)) #converting origin to factor

# Plotting matrix of pair-wise scatterplots of all variables
ggpairs(select(Auto, -name)) # not including name because it has too many categories
```




```
ggplot(data = Auto) +
  geom_bar(mapping = aes(x = name)) # bar chart of name. Not very useful because there are so many cate
```



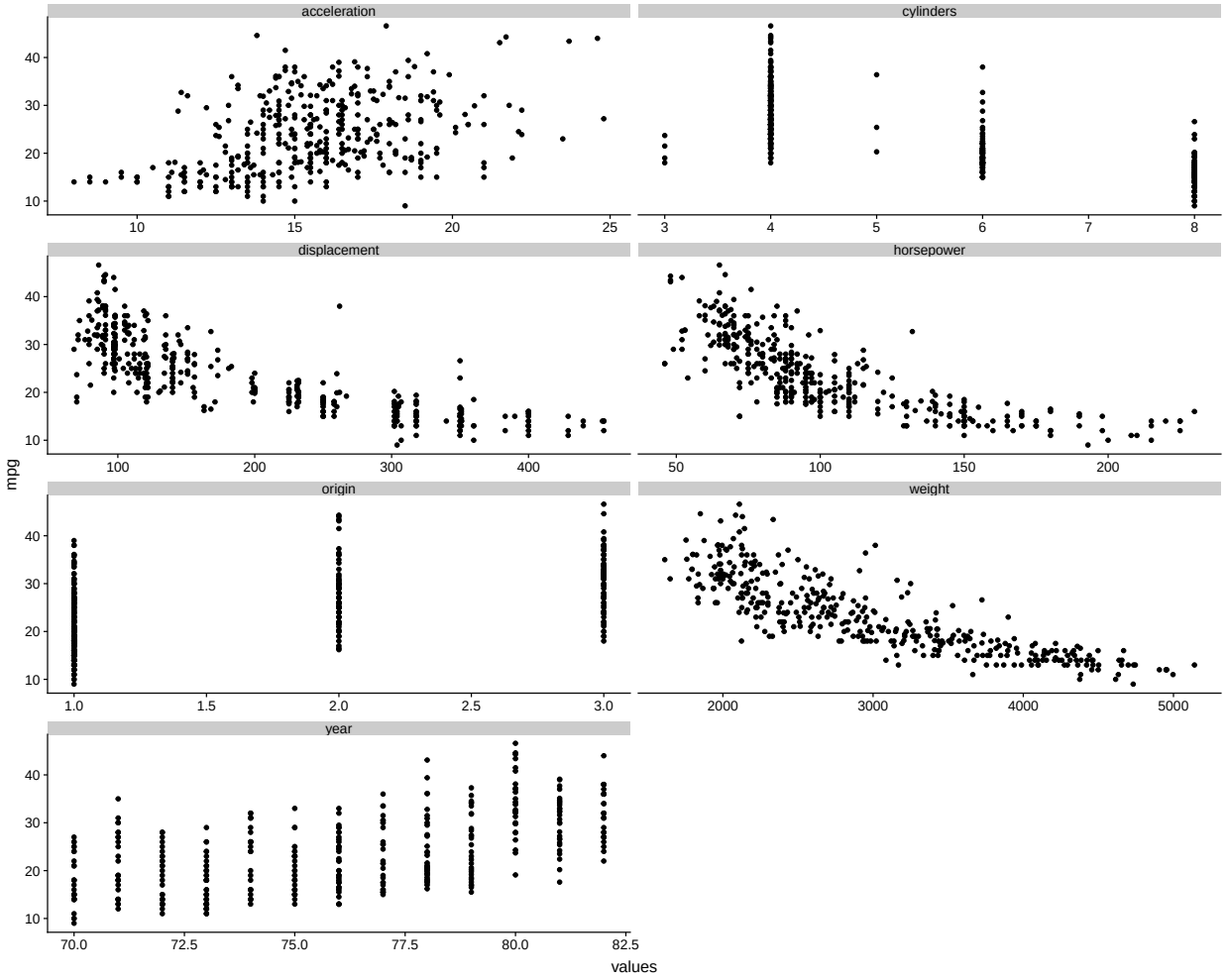
Key points:

- Many of the variables are strongly correlated. For example, `weight` has a correlation coefficient with an absolute value higher than 0.8 with `mpg`, `cylinders`, `displacement` and `horsepower`.

- The distribution of `mpg` is slightly negatively skewed.
- There are some linear relationships between some variables. e.g. between: `cylinders` and `displacement`, `displacement` and `horsepower`, `displacement` and `weight`, `horsepower` and `weight`.
- Relationships between `year` and the other variables are very noisy, although to a lesser extent for `mpg` and `acceleration`.
- `origin` affects `mpg` (e.g. higher `mpg` for `origin = 3`), but that could be because of changes in `origin` over time (because `year` also affects `mpg`) rather than a direct effect of `origin` on `mpg`.

(f) Suppose that we wish to predict gas mileage (mpg) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting mpg? Justify your answer.

```
# suppressing messages from readr
options(readr.num_columns = 0)

Auto <- read_csv("Auto.csv", na = c("", "NA", "?")) # loading data, making sure missing values (?s) are
Auto <- na.omit(Auto) # removing missing values
# Not converting characters and origin to factors because this causes problems with `gather`

Auto_wrap <- gather(select(Auto, -name), -mpg, key = "vars", value = "values") # getting data in the ri

  ggplot(Auto_wrap, aes(x = values, y = mpg)) +
    geom_point() +
    facet_wrap(~ vars, scales = "free_x", ncol = 2)
```



All of the variables except `name` look like they would be useful for predicting, with `displacement`, `horsepower`, `weight` and `cylinder` looking particularly useful because of their strong correlations with `mpg` (see above figure and correlation coefficients in 9(e)). `name` is not very useful (no clear relationship with `mpg` and it would introduce way too many dummy variables).

Exercise 10

(a) To begin, load in the `Boston` data set. The Boston data set is part of the `MASS` library in R. How many rows are in this data set? How many columns? What do the rows and columns represent?

```
library (MASS)
# Now the data set is contained in the object Boston.
as_tibble(Boston)
```

```
## # A tibble: 506 x 14
##       crim    zn indus  chas   nox    rm   age   dis   rad   tax ptratio
##      <dbl> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <dbl> <int> <dbl>   <dbl>
##  1 0.00632  18    2.31     0 0.538  6.58  65.2  4.09     1   296    15.3
##  2 0.0273    0    7.07     0 0.469  6.42  78.9  4.97     2   242    17.8
##  3 0.0273    0    7.07     0 0.469  7.18  61.1  4.97     2   242    17.8
##  4 0.0324    0    2.18     0 0.458  7.00  45.8  6.06     3   222    18.7
```

```
##  5 0.0690    0    2.18     0 0.458  7.15  54.2  6.06     3   222    18.7
##  6 0.0298    0    2.18     0 0.458  6.43  58.7  6.06     3   222    18.7
##  7 0.0883   12.5  7.87     0 0.524  6.01  66.6  5.56     5   311    15.2
##  8 0.145    12.5  7.87     0 0.524  6.17  96.1  5.95     5   311    15.2
##  9 0.211    12.5  7.87     0 0.524  5.63 100    6.08     5   311    15.2
## 10 0.170    12.5  7.87     0 0.524  6.00  85.9  6.59     5   311    15.2
## # ... with 496 more rows, and 3 more variables: black <dbl>, lstat <dbl>,
## #   medv <dbl>
```

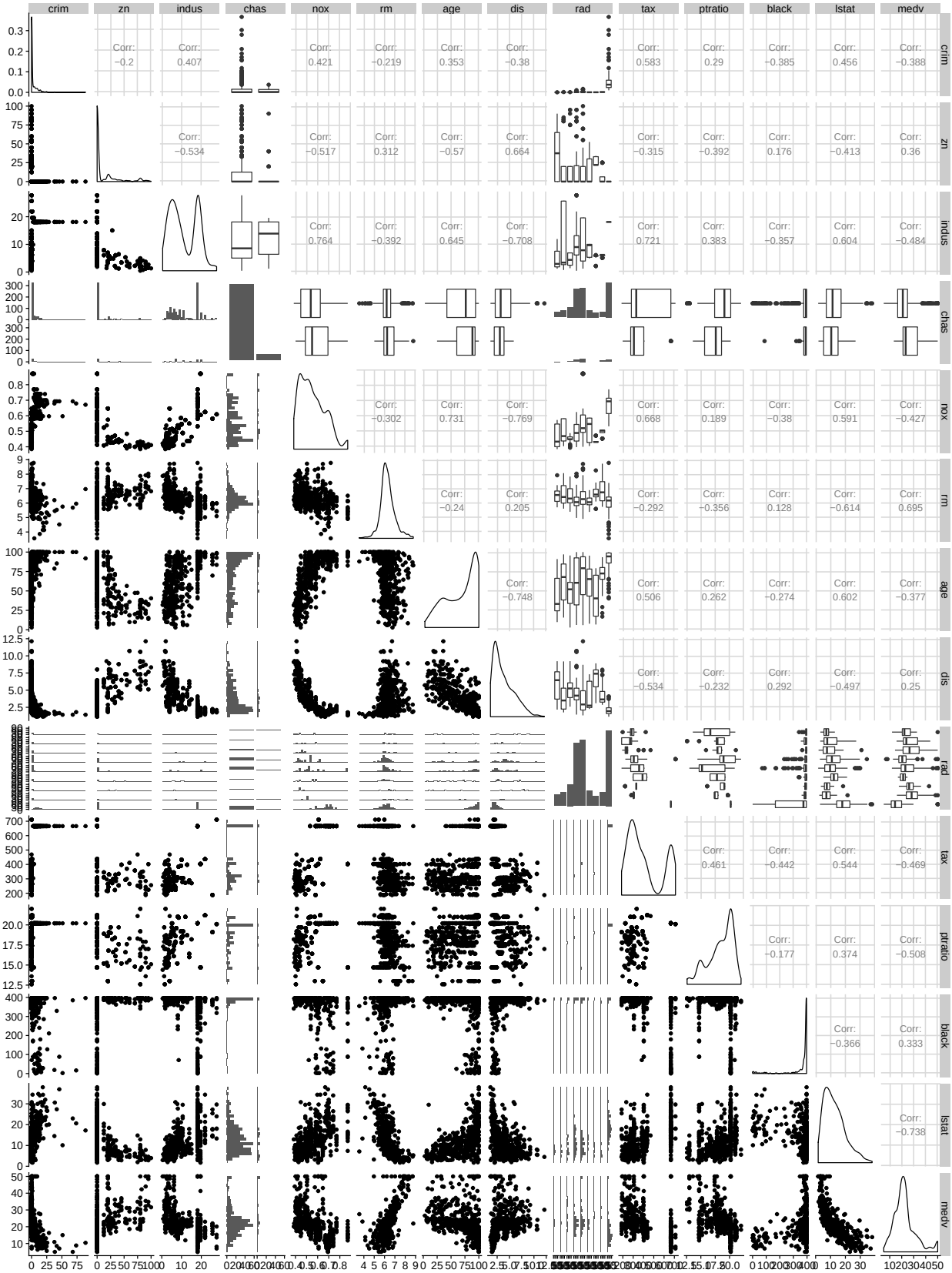
```
#Read about the data set:
#?Boston
```

The data set has 506 rows and 14 columns. The rows represent towns. The columns represent the following variables:

- `crim`: per capita crime rate by town.
- `zn`: proportion of residential land zoned for lots over 25,000 sq.ft.
- `indus`: proportion of non-retail business acres per town.
- `chas`: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- `nox`: nitrogen oxides concentration (parts per 10 million).
- `rm`: average number of rooms per dwelling.
- `age`: proportion of owner-occupied units built prior to 1940.
- `dis`: weighted mean of distances to five Boston employment centres.
- `rad`: index of accessibility to radial highways.
- `tax`: full-value property-tax rate per $10,000.
- `ptratio`: pupil-teacher ratio by town.
- `black`: 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town.
- `lstat`: lower status of the population (percent).
- `medv`: median value of owner-occupied homes in $1000s.

(b) Make some pairwise scatterplots of the predictors (columns) in this data set. Describe your findings.

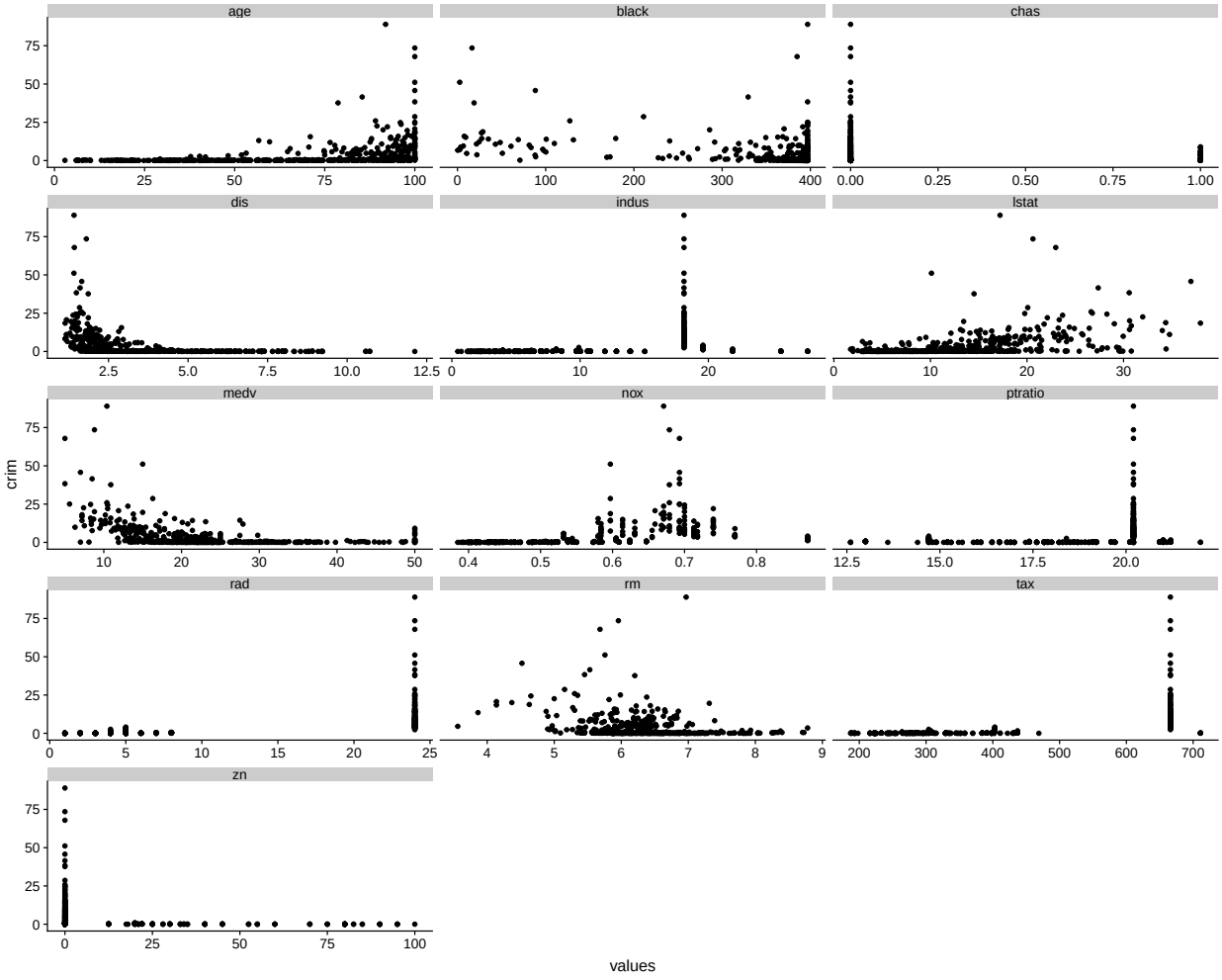
```
Boston_10b <- as_tibble(Boston) %>%
  mutate_if(is.integer, factor) # converting integers to factors

# Plotting matrix of pair-wise scatterplots of all variables
ggpairs(Boston_10b)
```



(c) Are any of the predictors associated with per capita crime rate? If so, explain the rela-

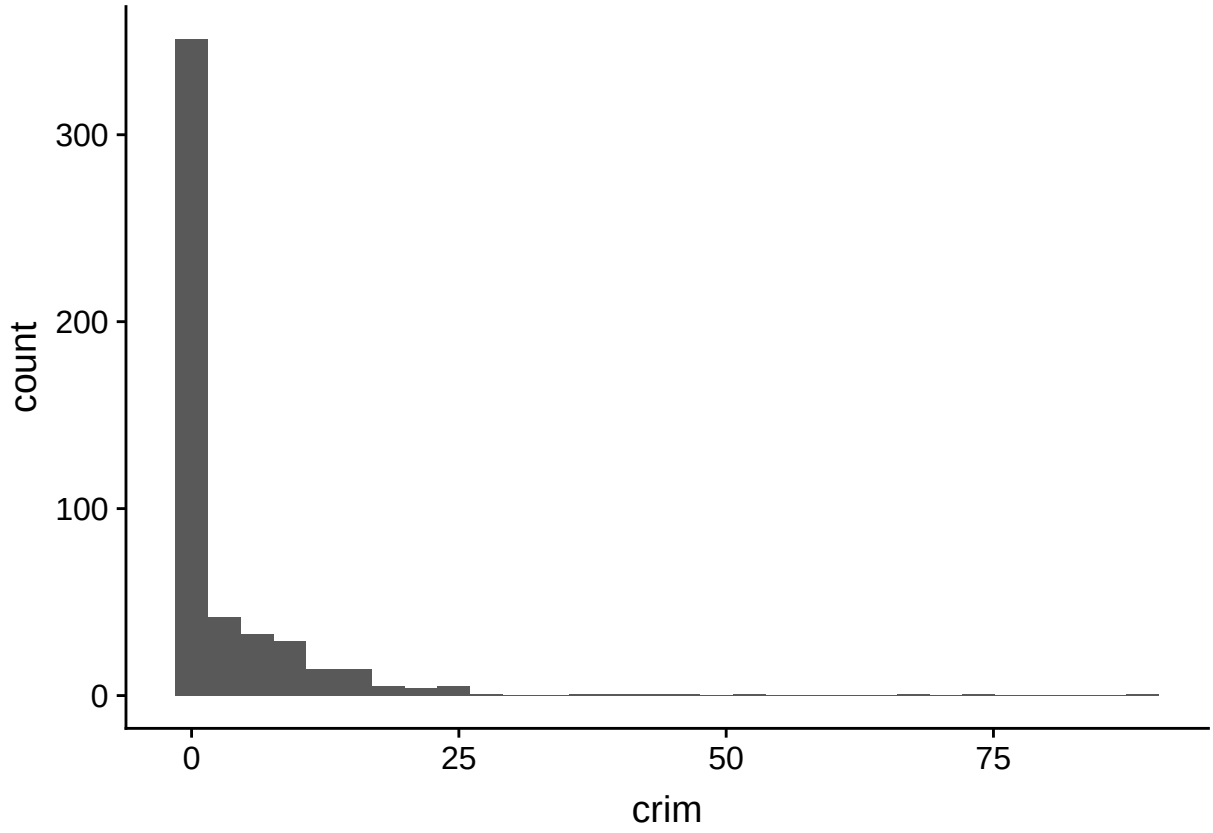
tionship.

```
Boston_wrap <- as_tibble(Boston) %>%
  gather(-crim, key = "vars", value = "values") # getting data in the right format to use for facet_wra

  ggplot(Boston_wrap, aes(x = values, y = crim)) +
    geom_point() +
    facet_wrap(~ vars, scales = "free_x", ncol = 3)
```

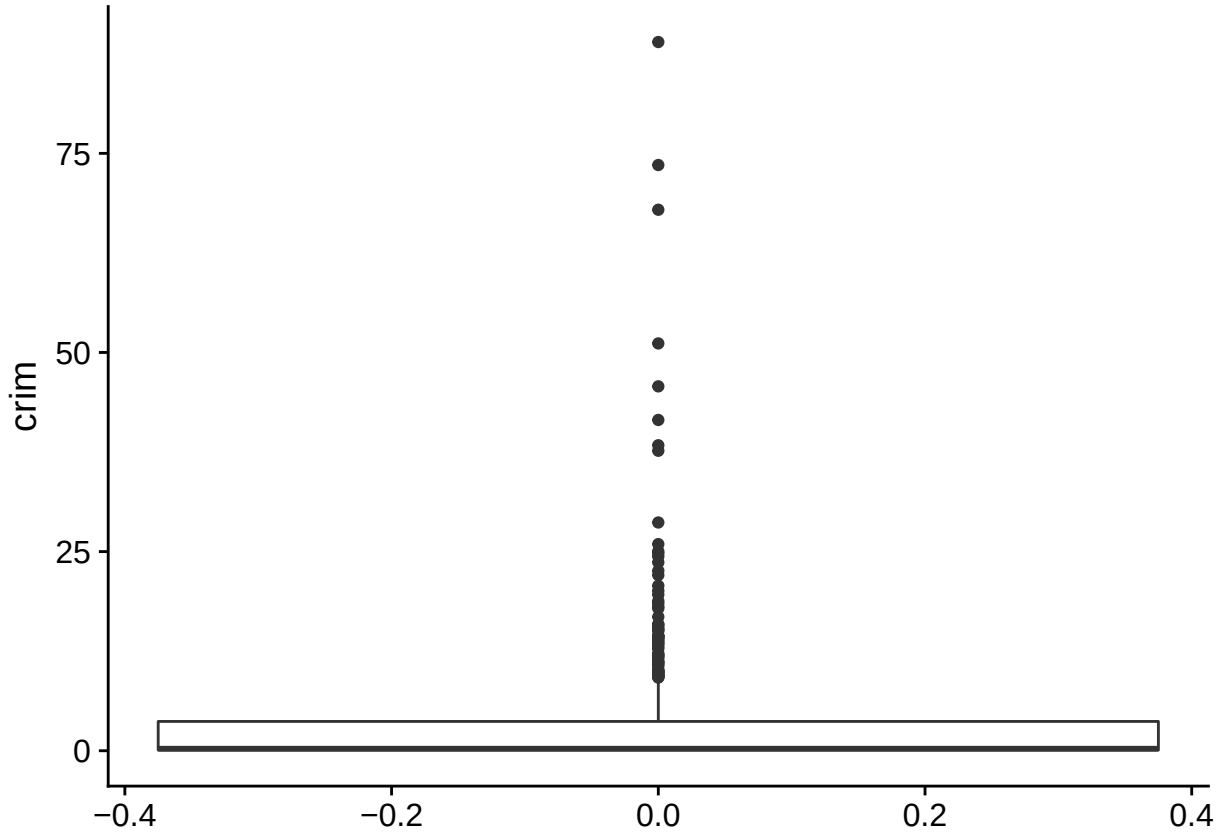


(d) Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.

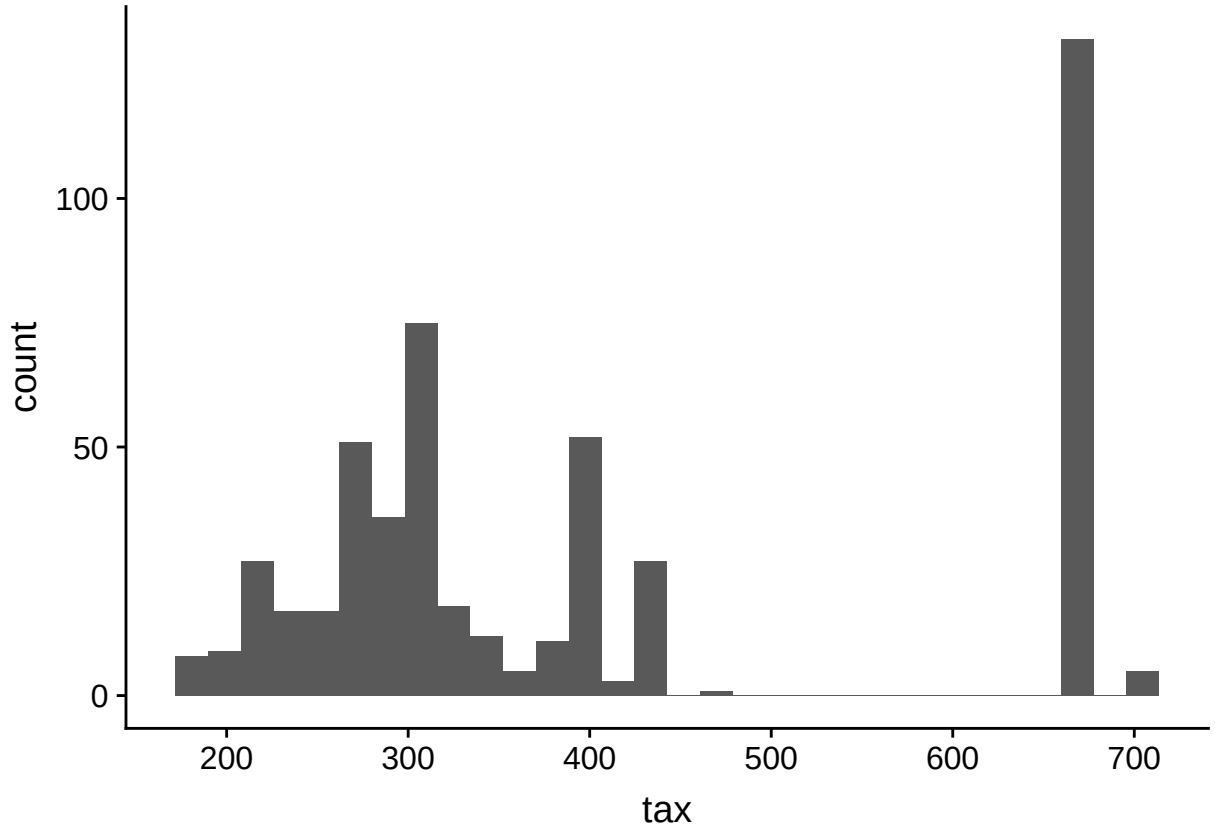
```
Boston_10d <- as_tibble(Boston) %>%
  mutate_if(is.integer, factor)# converting integers to factors

ggplot(Boston_10d) +
  geom_histogram(mapping = aes(x = crim))
```



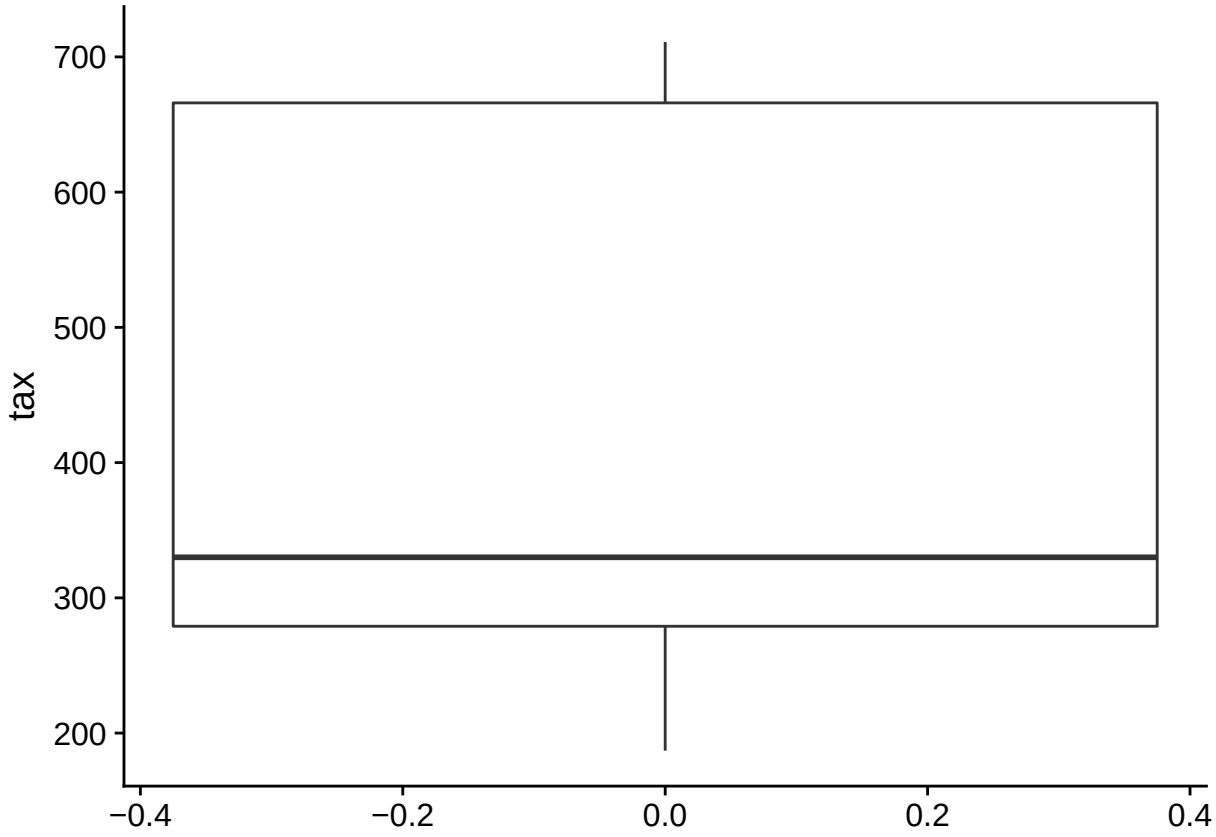

```
ggplot(Boston_10d, mapping = aes(y = crim)) +
  geom_boxplot()
```



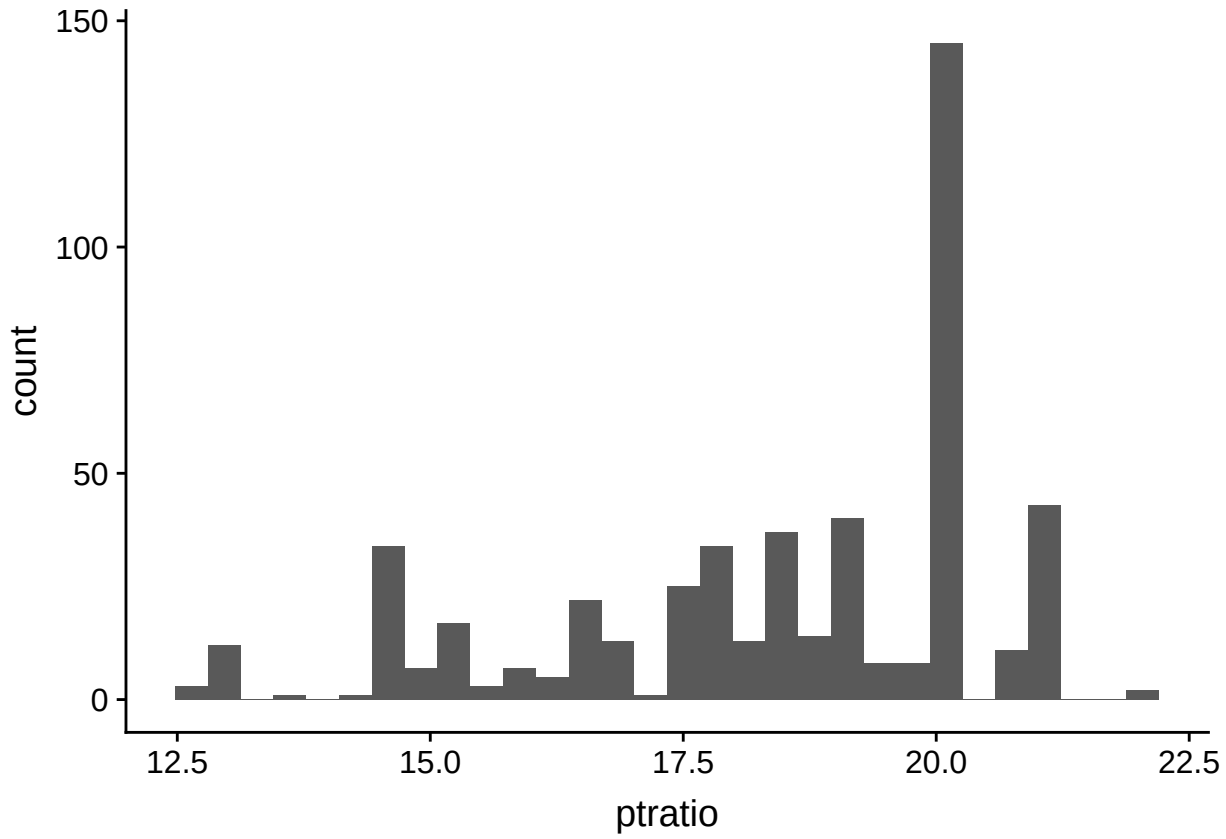

```
ggplot(Boston_10d) +
  geom_histogram(mapping = aes(x = tax))
```



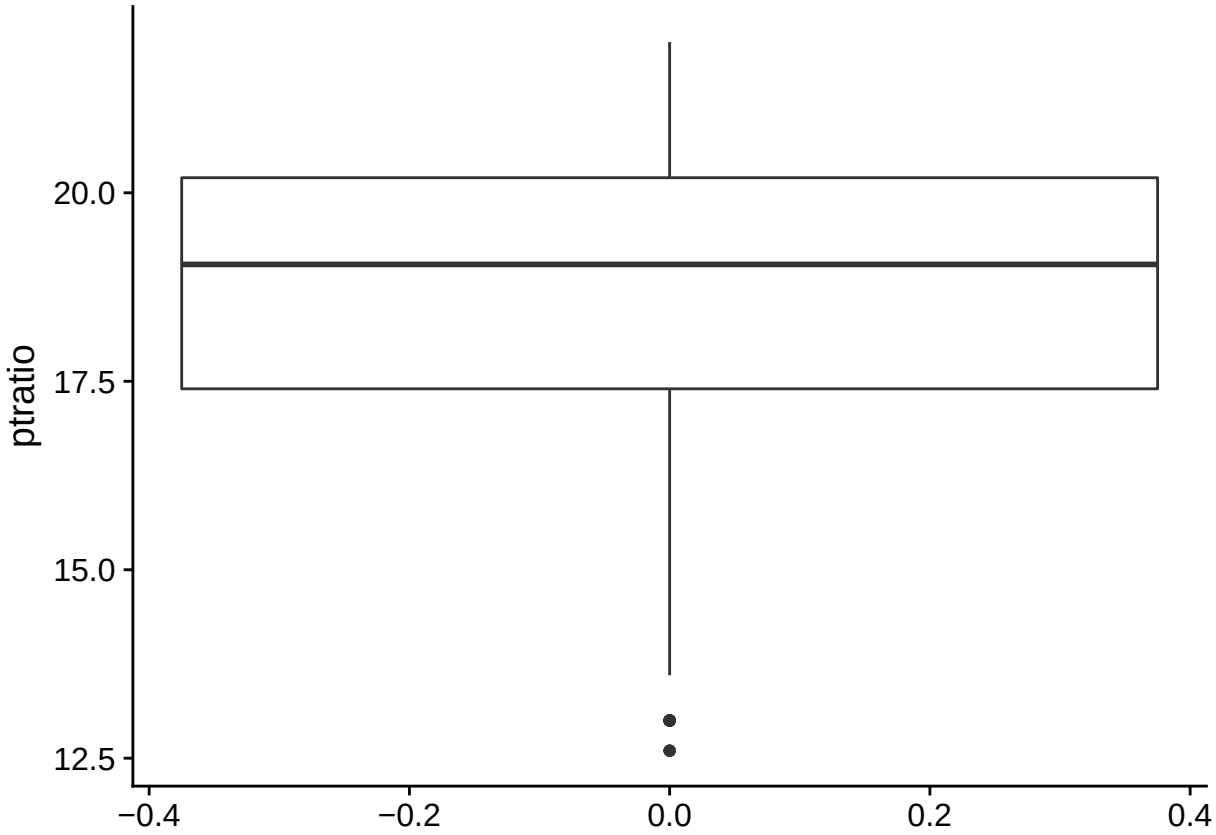

```
ggplot(Boston_10d, mapping = aes(y = tax)) +
  geom_boxplot()
```




```
ggplot(Boston_10d) +
  geom_histogram(mapping = aes(x = ptratio))
```




```
ggplot(Boston_10d, mapping = aes(y = ptratio)) +
  geom_boxplot()
```




```
skim(Boston_10d)
```

```
## Skim summary statistics
##  n obs: 506
##  n variables: 14
##
## -- Variable type:factor ------------------------------------------------------------------------
##  variable missing complete   n n_unique                    top_counts
##      chas       0      506 506        2           0: 471, 1: 35, NA: 0
##       rad       0      506 506        9 24: 132, 5: 115, 4: 110, 3: 38
##  ordered
##    FALSE
##    FALSE
##
## -- Variable type:numeric -----------------------------------------------------------------------
##  variable missing complete   n   mean     sd       p0     p25    p50
##       age       0      506 506  68.57  28.15   2.9     45.02   77.5
##     black       0      506 506 356.67  91.29   0.32   375.38  391.44
##      crim       0      506 506   3.61   8.6    0.0063   0.082   0.26
##       dis       0      506 506   3.8    2.11   1.13     2.1     3.21
##     indus       0      506 506  11.14   6.86   0.46     5.19    9.69
##     lstat       0      506 506  12.65   7.14   1.73     6.95   11.36
##      medv       0      506 506  22.53   9.2    5       17.02   21.2
##       nox       0      506 506   0.55   0.12   0.38     0.45    0.54
##   ptratio       0      506 506  18.46   2.16  12.6     17.4    19.05
##        rm       0      506 506   6.28   0.7    3.56     5.89    6.21
```

```
##       tax       0      506 506 408.24 168.54 187      279      330
##        zn       0      506 506  11.36  23.32   0        0        0
##     p75   p100     hist
##   94.07 100    <U+2581><U+2582><U+2582><U+2582><U+2582><U+2582><U+2583><U+2587>
##  396.23 396.9  <U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2587>
##    3.68  88.98 <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##    5.19  12.13 <U+2587><U+2585><U+2583><U+2583><U+2582><U+2581><U+2581><U+2581>
##   18.1   27.74 <U+2583><U+2586><U+2585><U+2581><U+2581><U+2587><U+2581><U+2581>
##   16.96  37.97 <U+2586><U+2587><U+2586><U+2585><U+2582><U+2581><U+2581><U+2581>
##   25     50    <U+2582><U+2585><U+2587><U+2586><U+2582><U+2582><U+2581><U+2581>
##    0.62   0.87 <U+2587><U+2586><U+2587><U+2586><U+2583><U+2585><U+2581><U+2581>
##   20.2   22    <U+2581><U+2582><U+2582><U+2582><U+2585><U+2585><U+2587><U+2583>
##    6.62   8.78 <U+2581><U+2581><U+2582><U+2587><U+2587><U+2582><U+2581><U+2581>
##  666    711    <U+2583><U+2587><U+2582><U+2585><U+2581><U+2581><U+2581><U+2586>
##   12.5  100    <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
```

(e) How many of the suburbs in this data set bound the Charles river?

35 suburbs bound the Charles River.

(f) What is the median pupil-teacher ratio among the towns in this data set?

Median = 19.05

(g) Which suburb of Boston has lowest median value of owneroccupied homes? What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

```
# suburb of Boston has lowest median value of owneroccupied homes
which(Boston$medv == min(Boston$medv))
```

```
## [1] 399 406
```

```
# values of predictors for suburbs with the lowest median value of owneroccupied homes
kable(filter(Boston, medv == min(medv)))
```

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
38.3518	0	18.1	0	0.693	5.453	100	1.4896	24	666	20.2	396.90	30.59	5
67.9208	0	18.1	0	0.693	5.683	100	1.4254	24	666	20.2	384.97	22.98	5

```
# Range calculation:
kable(max_Bost <- summarise_if(Boston, is.numeric, list(max))) # calculating max for all numeric variab
```

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
88.9762	100	27.74	1	0.871	8.78	100	12.1265	24	711	22	396.9	37.97	50

```
kable(min_Bost <- summarise_if(Boston, is.numeric, list(min))) # calculating min for all numeric variab
```

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.00632	0	0.46	0	0.385	3.561	2.9	1.1296	1	187	12.6	0.32	1.73	5

(h) In this data set, how many of the suburbs average more than seven rooms per dwelling? More than eight rooms per dwelling? Comment on the suburbs that average more than eight

rooms per dwelling.

```
# suburbs average more than seven rooms per dwelling
filter(Boston, rm > 7) %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    64
```

```
# suburbs average more than eight rooms per dwelling
filter(Boston, rm > 8) %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    13
```

```
kable(filter(Boston, rm > 8))
```

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.12083	0	2.89	0	0.4450	8.069	76.0	3.4952	2	276	18.0	396.90	4.21	38.7
1.51902	0	19.58	1	0.6050	8.375	93.9	2.1620	5	403	14.7	388.45	3.32	50.0
0.02009	95	2.68	0	0.4161	8.034	31.9	5.1180	4	224	14.7	390.55	2.88	50.0
0.31533	0	6.20	0	0.5040	8.266	78.3	2.8944	8	307	17.4	385.05	4.14	44.8
0.52693	0	6.20	0	0.5040	8.725	83.0	2.8944	8	307	17.4	382.00	4.63	50.0
0.38214	0	6.20	0	0.5040	8.040	86.5	3.2157	8	307	17.4	387.38	3.13	37.6
0.57529	0	6.20	0	0.5070	8.337	73.3	3.8384	8	307	17.4	385.91	2.47	41.7
0.33147	0	6.20	0	0.5070	8.247	70.4	3.6519	8	307	17.4	378.95	3.95	48.3
0.36894	22	5.86	0	0.4310	8.259	8.4	8.9067	7	330	19.1	396.90	3.54	42.8
0.61154	20	3.97	0	0.6470	8.704	86.9	1.8010	5	264	13.0	389.70	5.12	50.0
0.52014	20	3.97	0	0.6470	8.398	91.5	2.2885	5	264	13.0	386.86	5.91	48.8
0.57834	20	3.97	0	0.5750	8.297	67.0	2.4216	5	264	13.0	384.54	7.44	50.0
3.47428	0	18.10	1	0.7180	8.780	82.9	1.9047	24	666	20.2	354.55	5.29	21.9

New packages/functions

New packages/functions I've learnt during these exercises:

- `summarise_if`
- `slice`
- `gather`
- `case_when`
- use `as_tibble()` instead of `as.tibble()`