Machine Learning for Social Science

Introduction, R & RStudio

Setup

We start by first installing some packages that we will need throughout this notebook.

```r
# install.packages("tidyverse")
# install.packages("GGally")
# install.packages("mlbench")
```

Besides installing the packages, they also have to be loaded in order to be operational.

```r
#library(learnr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(mlbench)
```

R-Basics, help & packages

This section lists some useful functions when working with R. First of all, it is good practice to cite R whenever it was used in the research process. `citation()` displays the proper way to cite R, whereas `citation("packagename")` can be used when citing R packages.

```r
citation()
```

```
## To cite R in publications use:
##
##   R Core Team (2024). _R: A Language and Environment for Statistical
##   Computing_. R Foundation for Statistical Computing, Vienna, Austria.
##   <https://www.R-project.org/>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {R: A Language and Environment for Statistical Computing},
##     author = {{R Core Team}},
##     organization = {R Foundation for Statistical Computing},
##     address = {Vienna, Austria},
##     year = {2024},
##     url = {https://www.R-project.org/},
##   }
##
## We have invested a lot of time and effort in creating R, please cite it
## when using it for data analysis. See also ’citation("pkgname")’ for
## citing R packages.
```

```r
citation("ggplot2")
```

```
## To cite ggplot2 in publications, please use
##
##   H. Wickham. ggplot2: Elegant Graphics for Data Analysis.
##   Springer-Verlag New York, 2016.
##
## A BibTeX entry for LaTeX users is
##
##   @Book{,
##     author = {Hadley Wickham},
##     title = {ggplot2: Elegant Graphics for Data Analysis},
##     publisher = {Springer-Verlag New York},
##     year = {2016},
##     isbn = {978-3-319-24277-4},
##     url = {https://ggplot2.tidyverse.org},
##   }
```

Typically, one of the first things to do is specifying your working directory. The following functions can be used to display (`getwd()`) and set (`setwd()`) the working directory and to list its contents (`dir()`). Keep in mind that R only accepts paths with forward slashes.

```r
getwd()
```

```
## [1] "/home/kevin/repos/UMD_classes_code/ML_social_science_SURV613/class_labs"
```

```r
# setwd("path")
dir()
```

```
##  [1] "introduction_files"             "introduction.html"
##  [3] "introduction.pdf"               "introduction.Rmd"
##  [5] "knn_files"                       "knn.html"
##  [7] "knn.qmd"                         "ml-basics_files"
##  [9] "ml-basics.html"                  "ml-basics.Rmd"
## [11] "regularized-regression-1_files" "regularized-regression-1.html"
## [13] "regularized-regression-1.Rmd"   "regularized-regression-2_files"
## [15] "regularized-regression-2.html"  "regularized-regression-2.Rmd"
## [17] "Rplots.pdf"
```

To get familiar with R’s help system, we can explore the documentation for the function `help()`. This is equivalent to `help(help)`

```r
help()
```

The documentation for global R options.

```r
help(options)
```

Use `help.search()` to search the help system.

```r
help.search("glm")
```

Working with data

In this notebook, we use the Boston Housing data set. “This dataset contains information collected by the U.S Census Service concerning housing in the area of Boston Mass. It was obtained from the StatLib archive (http://lib.stat.cmu.edu/datasets/boston), and has been used extensively throughout the literature to benchmark algorithms.”

Source: https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html

```r
data(BostonHousing2)
boston <- BostonHousing2
```

As a shortcut for `help()` we can use `?` to get some information about this dataset.

```r
?BostonHousing2
```

The following functions can be used to get a first impression of the data.

```r
str(boston)
```

```
## ’data.frame':    506 obs. of  19 variables:
##  $ town   : Factor w/ 92 levels "Arlington","Ashland",..: 54 77 77 46 46 46 69 69 69 69 ...
##  $ tract  : int  2011 2021 2022 2031 2032 2033 2041 2042 2043 2044 ...
##  $ lon    : num  -71 -71 -70.9 -70.9 -70.9 ...
##  $ lat    : num  42.3 42.3 42.3 42.3 42.3 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
##  $ cmedv  : num  24 21.6 34.7 33.4 36.2 28.7 22.9 22.1 16.5 18.9 ...
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : int  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ b      : num  397 397 393 395 397 ...
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
```

```
head(boston)
```

```
##         town tract      lon     lat medv cmedv    crim zn indus chas   nox
## 1     Nahant  2011 -70.9550 42.2550 24.0  24.0 0.00632 18  2.31    0 0.538
## 2 Swampscott  2021 -70.9500 42.2875 21.6  21.6 0.02731  0  7.07    0 0.469
## 3 Swampscott  2022 -70.9360 42.2830 34.7  34.7 0.02729  0  7.07    0 0.469
## 4 Marblehead  2031 -70.9280 42.2930 33.4  33.4 0.03237  0  2.18    0 0.458
## 5 Marblehead  2032 -70.9220 42.2980 36.2  36.2 0.06905  0  2.18    0 0.458
## 6 Marblehead  2033 -70.9165 42.3040 28.7  28.7 0.02985  0  2.18    0 0.458
##      rm  age    dis rad tax ptratio      b lstat
## 1 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
## 2 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
## 3 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
## 4 6.998 45.8 6.0622   3 222    18.7 394.63  2.94
## 5 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
## 6 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
```

Note that we can use `View()`, i.e. the data viewer, in connection with conditions on rows and columns to display only certain pieces of the whole data frame.

```
View(boston)
View(boston[-(1:500), 1:2])
```

Using index notation to access only specific variables or observations is an important tool as it can be used in conjunction with many different functions. It is therefore worthwhile to consider some basic examples.

```
# boston[, 1]
# boston[, 1:5]
# boston[1:10, c(1:2,5)]
```

List all variable names of the Boston Housing data.

```
names(boston)
```

```
##  [1] "town"    "tract"   "lon"     "lat"     "medv"    "cmedv"   "crim"   
##  [8] "zn"      "indus"   "chas"    "nox"     "rm"      "age"     "dis"    
## [15] "rad"     "tax"     "ptratio" "b"       "lstat"
```

Now we can access variables by using their names and the $-notation. This can be combined with conditional statements regarding rows to also filter specific observations.

```r
boston$medv
```

```
##   [1] 24.0 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 15.0 18.9 21.7 20.4 18.2
##  [16] 19.9 23.1 17.5 20.2 18.2 13.6 19.6 15.2 14.5 15.6 13.9 16.6 14.8 18.4 21.0
##  [31] 12.7 14.5 13.2 13.1 13.5 18.9 20.0 21.0 24.7 30.8 34.9 26.6 25.3 24.7 21.2
##  [46] 19.3 20.0 16.6 14.4 19.4 19.7 20.5 25.0 23.4 18.9 35.4 24.7 31.6 23.3 19.6
##  [61] 18.7 16.0 22.2 25.0 33.0 23.5 19.4 22.0 17.4 20.9 24.2 21.7 22.8 23.4 24.1
##  [76] 21.4 20.0 20.8 21.2 20.3 28.0 23.9 24.8 22.9 23.9 26.6 22.5 22.2 23.6 28.7
##  [91] 22.6 22.0 22.9 25.0 20.6 28.4 21.4 38.7 43.8 33.2 27.5 26.5 18.6 19.3 20.1
## [106] 19.5 19.5 20.4 19.8 19.4 21.7 22.8 18.8 18.7 18.5 18.3 21.2 19.2 20.4 19.3
## [121] 22.0 20.3 20.5 17.3 18.8 21.4 15.7 16.2 18.0 14.3 19.2 19.6 23.0 18.4 15.6
## [136] 18.1 17.4 17.1 13.3 17.8 14.0 14.4 13.4 15.6 11.8 13.8 15.6 14.6 17.8 15.4
## [151] 21.5 19.6 15.3 19.4 17.0 15.6 13.1 41.3 24.3 23.3 27.0 50.0 50.0 50.0 22.7
## [166] 25.0 50.0 23.8 23.8 22.3 17.4 19.1 23.1 23.6 22.6 29.4 23.2 24.6 29.9 37.2
## [181] 39.8 36.2 37.9 32.5 26.4 29.6 50.0 32.0 29.8 34.9 37.0 30.5 36.4 31.1 29.1
## [196] 50.0 33.3 30.3 34.6 34.9 32.9 24.1 42.3 48.5 50.0 22.6 24.4 22.5 24.4 20.0
## [211] 21.7 19.3 22.4 28.1 23.7 25.0 23.3 28.7 21.5 23.0 26.7 21.7 27.5 30.1 44.8
## [226] 50.0 37.6 31.6 46.7 31.5 24.3 31.7 41.7 48.3 29.0 24.0 25.1 31.5 23.7 23.3
## [241] 22.0 20.1 22.2 23.7 17.6 18.5 24.3 20.5 24.5 26.2 24.4 24.8 29.6 42.8 21.9
## [256] 20.9 44.0 50.0 36.0 30.1 33.8 43.1 48.8 31.0 36.5 22.8 30.7 50.0 43.5 20.7
## [271] 21.1 25.2 24.4 35.2 32.4 32.0 33.2 33.1 29.1 35.1 45.4 35.4 46.0 50.0 32.2
## [286] 22.0 20.1 23.2 22.3 24.8 28.5 37.3 27.9 23.9 21.7 28.6 27.1 20.3 22.5 29.0
## [301] 24.8 22.0 26.4 33.1 36.1 28.4 33.4 28.2 22.8 20.3 16.1 22.1 19.4 21.6 23.8
## [316] 16.2 17.8 19.8 23.1 21.0 23.8 23.1 20.4 18.5 25.0 24.6 23.0 22.2 19.3 22.6
## [331] 19.8 17.1 19.4 22.2 20.7 21.1 19.5 18.5 20.6 19.0 18.7 32.7 16.5 23.9 31.2
## [346] 17.5 17.2 23.1 24.5 26.6 22.9 24.1 18.6 30.1 18.2 20.6 17.8 21.7 22.7 22.6
## [361] 25.0 19.9 20.8 16.8 21.9 27.5 21.9 23.1 50.0 50.0 50.0 50.0 50.0 13.8 13.8
## [376] 15.0 13.9 13.3 13.1 10.2 10.4 10.9 11.3 12.3  8.8  7.2 10.5  7.4 10.2 11.5
## [391] 15.1 23.2  9.7 13.8 12.7 13.1 12.5  8.5  5.0  6.3  5.6  7.2 12.1  8.3  8.5
## [406]  5.0 11.9 27.9 17.2 27.5 15.0 17.2 17.9 16.3  7.0  7.2  7.5 10.4  8.8  8.4
## [421] 16.7 14.2 20.8 13.4 11.7  8.3 10.2 10.9 11.0  9.5 14.5 14.1 16.1 14.3 11.7
## [436] 13.4  9.6  8.7  8.4 12.8 10.5 17.1 18.4 15.4 10.8 11.8 14.9 12.6 14.1 13.0
## [451] 13.4 15.2 16.1 17.8 14.9 14.1 12.7 13.5 14.9 20.0 16.4 17.7 19.5 20.2 21.4
## [466] 19.9 19.0 19.1 19.1 20.1 19.9 19.6 23.2 29.8 13.8 13.3 16.7 12.0 14.6 21.4
## [481] 23.0 23.7 25.0 21.8 20.6 21.2 19.1 20.6 15.2  7.0  8.1 13.6 20.1 21.8 24.5
## [496] 23.1 19.7 18.3 21.2 17.5 16.8 22.4 20.6 23.9 22.0 11.9
```

```r
boston$medv[1:10]
```

```
##  [1] 24.0 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9
```

```r
boston$medv[boston$chas == 1]
```

```
##  [1] 13.4 15.3 17.0 15.6 27.0 50.0 50.0 24.4 20.0 21.7 19.3 22.4 23.3 21.5 23.0
## [16] 26.7 21.7 27.5 29.0 25.1 20.7 35.2 32.4 33.2 33.1 46.0 50.0 17.8 21.7 22.7
## [31] 16.8 21.9 50.0 50.0 50.0
```

We can also draw random samples from our data set and store those in new objects.

```r
index <- sample(1:nrow(boston), 0.75*nrow(boston))
subset <- boston[index,]
nrow(subset)
```

```
## [1] 379
```

Finally, here is a `dplyr` approach at selecting rows and columns of the Boston housing dataset.

```
boston %>%
  select(medv, chas) %>%
  filter(chas == 1)
```

```
##     medv chas
## 143 13.4    1
## 153 15.3    1
## 155 17.0    1
## 156 15.6    1
## 161 27.0    1
## 163 50.0    1
## 164 50.0    1
## 209 24.4    1
## 210 20.0    1
## 211 21.7    1
## 212 19.3    1
## 213 22.4    1
## 217 23.3    1
## 219 21.5    1
## 220 23.0    1
## 221 26.7    1
## 222 21.7    1
## 223 27.5    1
## 235 29.0    1
## 237 25.1    1
## 270 20.7    1
## 274 35.2    1
## 275 32.4    1
## 277 33.2    1
## 278 33.1    1
## 283 46.0    1
## 284 50.0    1
## 357 17.8    1
## 358 21.7    1
## 359 22.7    1
## 364 16.8    1
## 365 21.9    1
## 370 50.0    1
## 371 50.0    1
## 373 50.0    1
```

Exploring data

Basic descriptive statistics can be computed using `summary()`.

```
summary(boston$medv)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00   17.02   21.20   22.53   25.00   50.00
```

Note that this function is class-sensitive, i.e. here we get a different output depending on the class of the respective object.

```r
class(boston$medv)
```

```
## [1] "numeric"
```

```r
summary(boston$town)
```

```
##               Arlington                 Ashland                 Bedford 
##                       7                       2                       2 
##                 Belmont                 Beverly Boston Allston-Brighton 
##                       8                       6                       8 
##         Boston Back Bay      Boston Beacon Hill      Boston Charlestown 
##                       6                       3                       6 
##       Boston Dorchester         Boston Downtown      Boston East Boston 
##                      11                       8                      12 
##     Boston Forest Hills        Boston Hyde Park         Boston Mattapan 
##                       7                       4                       6 
##        Boston North End          Boston Roxbury       Boston Savin Hill 
##                       2                      19                      23 
##     Boston South Boston     Boston West Roxbury               Braintree 
##                      13                       4                       8 
##               Brookline              Burlington               Cambridge 
##                      12                       4                      30 
##                  Canton                 Chelsea                Cohasset 
##                       3                       5                       1 
##                 Concord                 Danvers                  Dedham 
##                       3                       4                       5 
##                   Dover                 Duxbury                 Everett 
##                       1                       1                       7 
##              Framingham                Hamilton                 Hanover 
##                      10                       1                       1 
##                 Hingham                Holbrook                    Hull 
##                       2                       2                       1 
##               Lexington                 Lincoln                    Lynn 
##                       6                       1                      22 
##               Lynnfield                  Malden              Manchester 
##                       2                       9                       1 
##              Marblehead              Marshfield                Medfield 
##                       3                       2                       1 
##                 Medford                 Melrose               Middleton 
##                      11                       4                       1 
##                  Millis                  Milton                  Nahant 
##                       1                       4                       1 
##                  Natick                 Needham                  Newton 
##                       6                       5                      18 
##                 Norfolk           North Reading                 Norwell 
##                       1                       2                       1 
##                 Norwood                 Peabody                Pembroke 
##                       5                       9                       2 
##                  Quincy                Randolph                 Reading 
##                      12                       3                       4
##                   Revere                Rockland                   Salem
##                        8                       2                       7
##                   Sargus                Scituate                  Sharon
##                        4                       2                       3
##                 Sherborn              Somerville                Stoneham
##                        1                      15                       3
##                  Sudbury              Swampscott               Topsfield
##                        2                       2                       1
##                Wakefield                 Walpole                 Waltham
##                        4                       3                      11
##                Watertown                 Wayland               Wellesley
##                        4                       2                       4
##                   Wenham                  Weston                Westwood
##                        1                       2                       3
##                 Weymouth              Wilmington              Winchester
##                        8                       3                       5
##                 Winthrop                  Woburn
##                        5                       6
```

```r
class(boston$town)
```

```
## [1] "factor"
```

Some summary statistics for the value of owner-occupied homes grouped by the `chas` river indicator, now using `dplyr`.

```r
boston %>%
  group_by(chas) %>%
  summarise(mean(medv), var(medv), min(medv), max(medv))
```

```
## # A tibble: 2 x 5
##   chas  ‘mean(medv)‘ ‘var(medv)‘ ‘min(medv)‘ ‘max(medv)‘
##   <fct>        <dbl>       <dbl>       <dbl>       <dbl>
## 1 0             22.1        78.0           5          50
## 2 1             28.4       140.         13.4          50
```

Summary statistics again, now for selected towns.

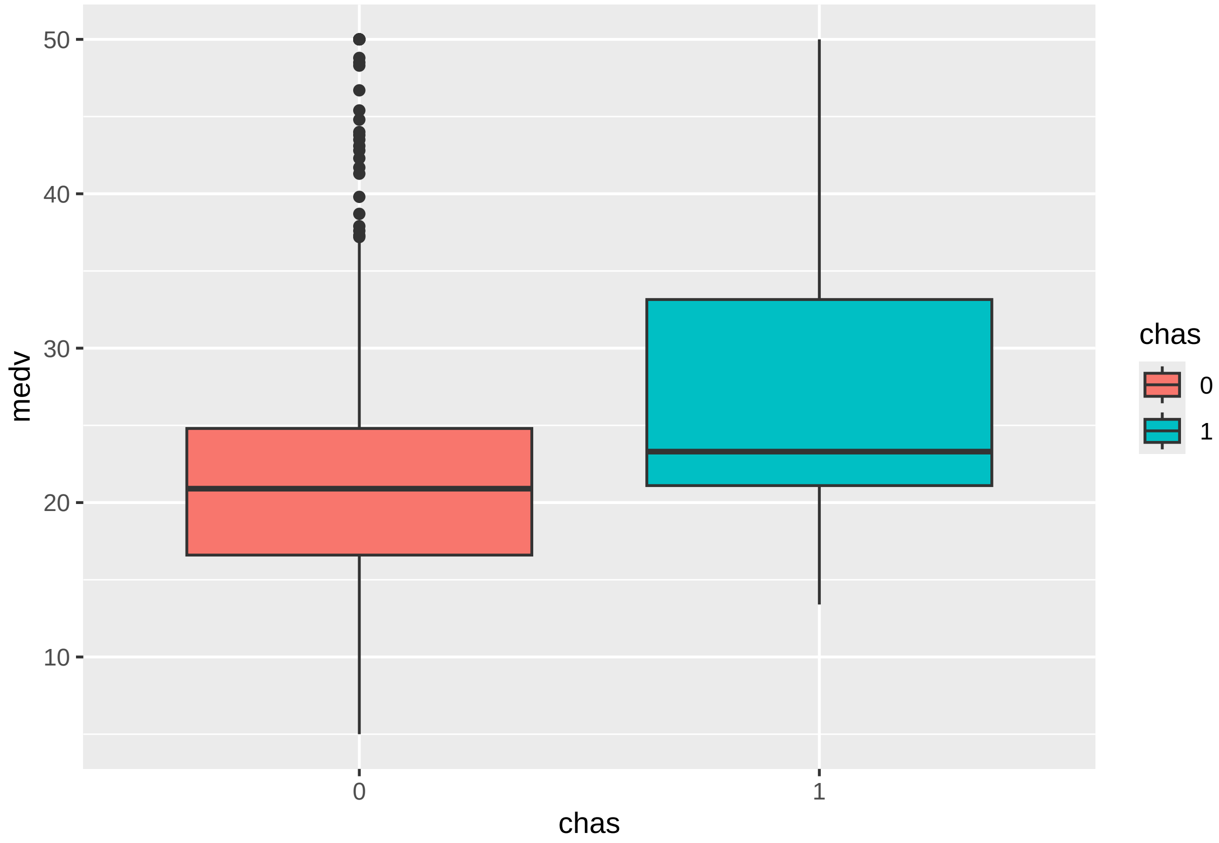
```r
boston %>%
  filter(town %in% c("Cambridge", "Boston South Boston")) %>%
  group_by(town) %>%
  summarise(mean = mean(medv), variance = var(medv), IQR = IQR(medv), n = n())
```

```
## # A tibble: 2 x 5
##   town                 mean variance   IQR     n
##   <fct>               <dbl>    <dbl> <dbl> <int>
## 1 Boston South Boston  9.12     10.9  6.2     13
## 2 Cambridge           23.6     143.   8.58    30
```

A boxplot via `qplot()`, separated by the `chas` dummy variable.

```r
qplot(chas, medv, data = boston, geom = "boxplot", fill = chas)
```

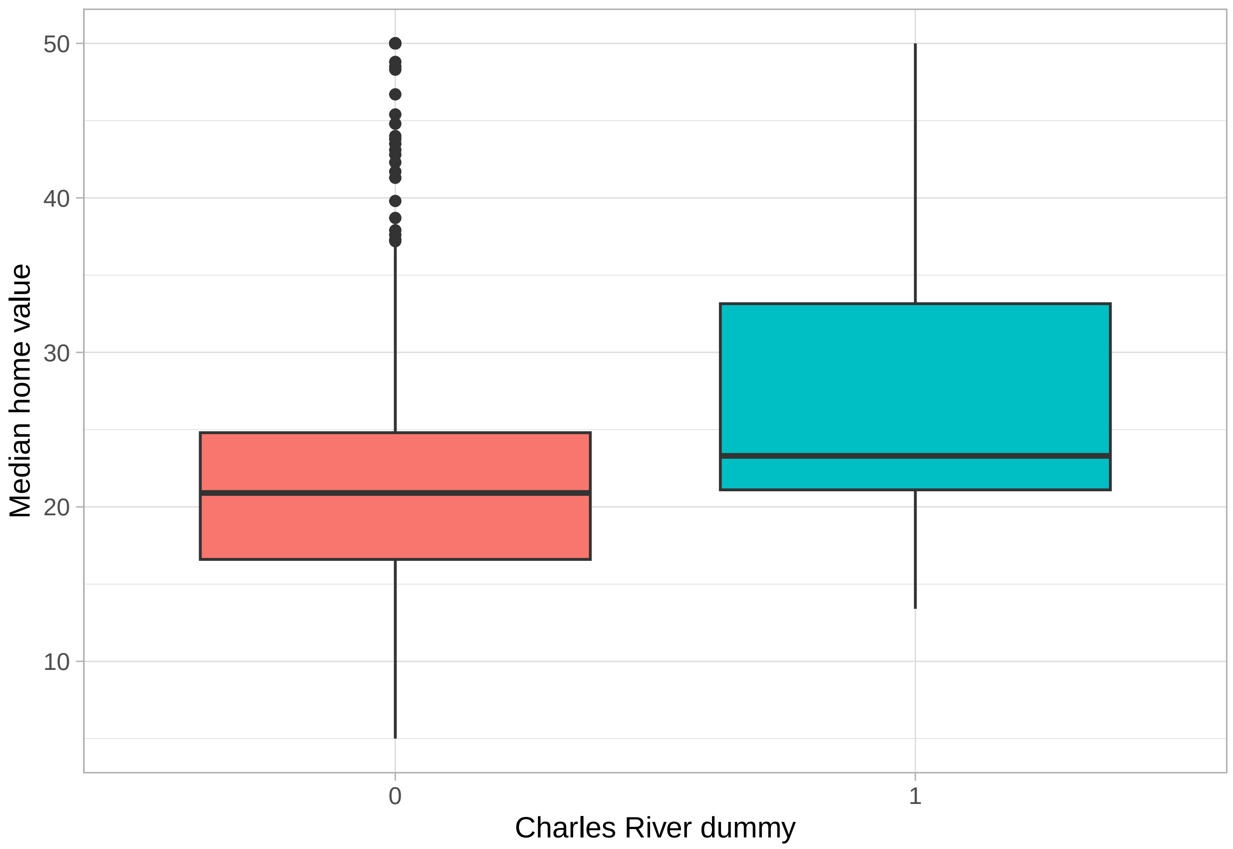
```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



The previous boxplot with better labels, now using the `ggplot()` function.

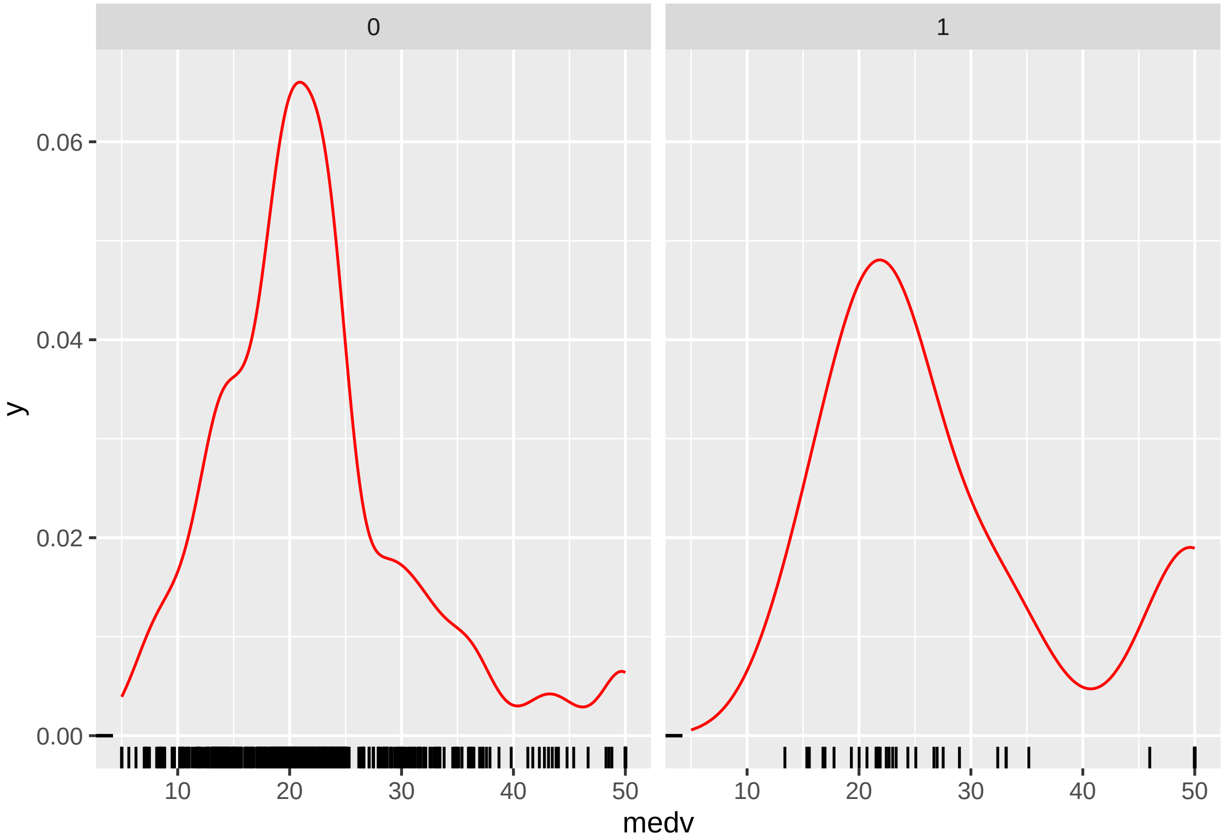
```r
ggplot(boston) +
  geom_boxplot(aes(x = chas, y = medv, fill = chas)) +
  labs(x = "Charles River dummy", y = "Median home value") +
  guides(fill = FALSE) +
  theme_light()
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



A density plot of the median value of owner-occupied homes, faceted by the river dummy.

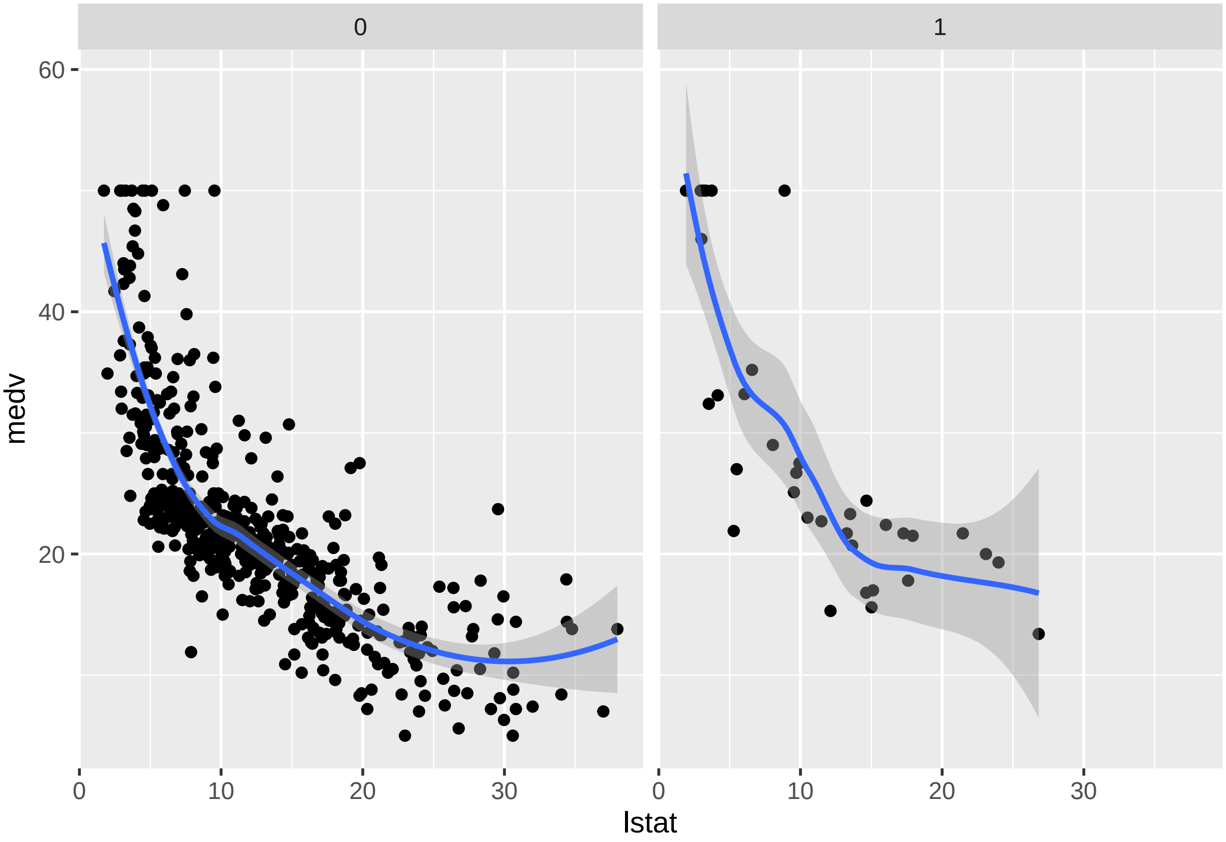
```
ggplot(boston) +
  geom_density(aes(x = medv), color = "red") +
  geom_rug(aes(x = medv, y = 0), position = position_jitter(height = 0)) +
  facet_grid(. ~ chas)
```



Grouped scatterplots of median home values and crime rates with overlayed loess curves.

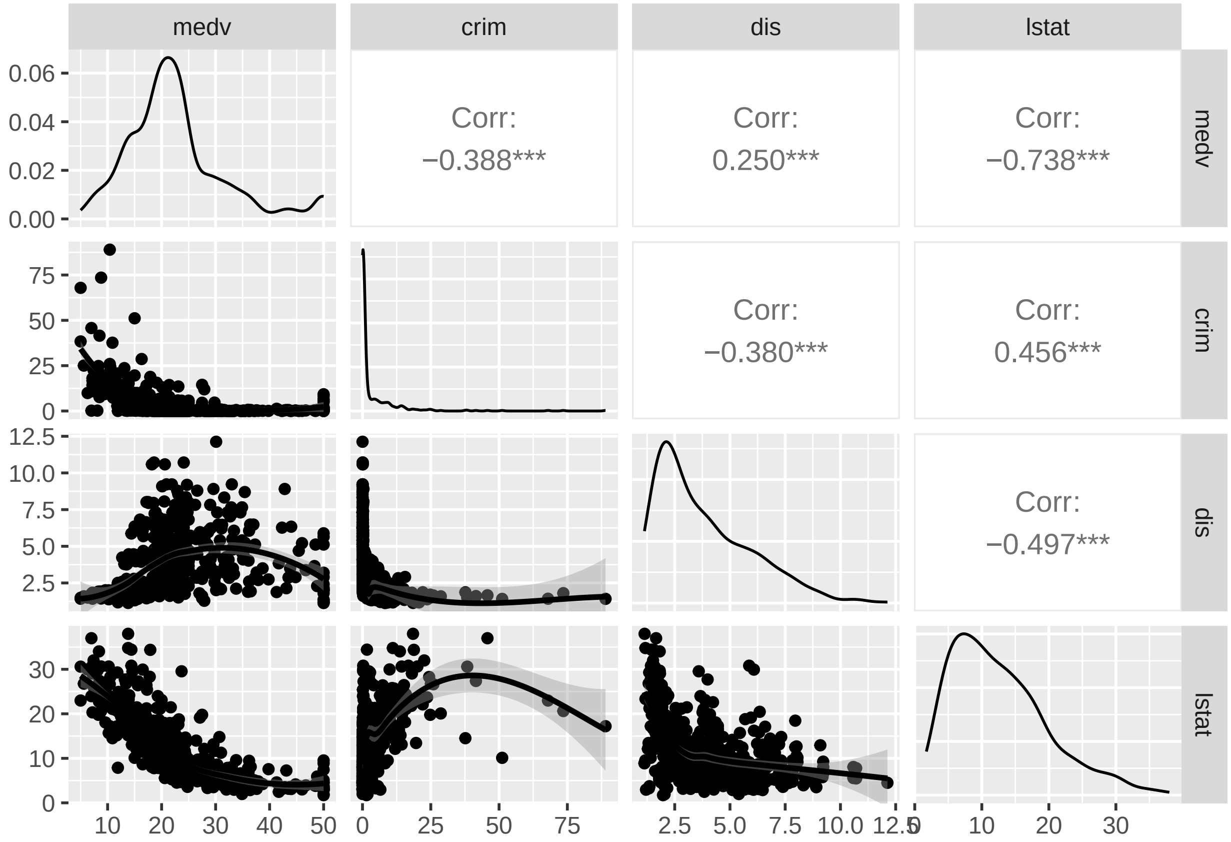
```
ggplot(boston) +
  geom_point(aes(x = lstat, y = medv)) +
  geom_smooth(aes(x = lstat, y = medv)) +
  facet_grid(. ~ chas)
```

```
## ‘geom_smooth()‘ using method = ’loess’ and formula = ’y ~ x’
```



Finally, a scatterplot matrix using `ggpairs()` from the `GGally` package.

```
ggpairs(boston[,c(5,7,14,19)], lower = list(continuous = "smooth_loess"))
```


medv
crim
dis
lstat
Corr:
−0.388***
Corr:
0.250***
Corr:
−0.738***
Corr:
−0.380***
Corr:
0.456***
Corr:
−0.497***
0.06
0.04
0.02
0.00
75
50
25
0
12.5
10.0
7.5
5.0
2.5
30
20
10
0
10
20
30
40
50
0
25
50
75
2.5
5.0
7.5
10.0
12.5
0
10
20
30