Homework 1

Rento Saijo

Department of Mathematics, Connecticut College

STA336: Statistical Machine Learning

Yan Zhuang, Ph.D.

January 30th, 2026

Disclosure

ChatGPT-5.2 was used to create the `YAML` portion to format the text nicely; I looked into the documentation for each of the package it used and added/removed unnecessary formattings. See the original `RMD` file here.

Problem 1

A very flexible approach has the advantage that it can represent a much wider range of possible shapes for f, and thus capture complicated (often non-linear) relationships between predictors X and a response Y. In contrast, a restrictive method like linear regression can only produce linear functions, e.g., $f(X) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$. The drawback of high flexibility is reduced interpretability: the fitted $\hat{f}$ can become so complex that it is difficult to understand how any individual predictor X_j is associated with Y, making flexible methods less attractive when inference and interpretability are the goal. Therefore, more flexible approaches are generally preferred when interpretability is not a priority and prediction is the primary objective, since we are willing to trade a clear description of predictor-response relationships for the ability to fit complex patterns; however, even for prediction, the most flexible model is not always best because highly flexible methods can overfit, so a less flexible method can sometimes yield better test performance. Conversely, a less flexible approach is preferred when inference is the goal because restrictive models are much more interpretable. *Source: ISLR2 §2.1.3, p. 24-6.*

Problem 2 (a)

```
# Load libraries.
suppressMessages(library(tidyverse))
suppressMessages(library(GGally))
suppressMessages(library(ISLR2))

# Load data.
data(Auto)

# Count missing values.
colSums(is.na(Auto)) # It seems that we have no missing values.
```

```
##          mpg    cylinders displacement   horsepower       weight acceleration
##            0            0            0            0            0            0
##         year       origin         name
```

```
##              0             0             0
```

```r
# Check structure.
tibble::glimpse(Auto)
```

```
## Rows: 392
## Columns: 9
## $ mpg          <dbl> 18, 15, 18, 16, 17, 15, 14, 14, 14, 15, 15, 14, 15, 14, 2~
## $ cylinders    <int> 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 4, 6, 6, 6, 4, ~
## $ displacement <dbl> 307, 350, 318, 304, 302, 429, 454, 440, 455, 390, 383, 34~
## $ horsepower   <int> 130, 165, 150, 150, 140, 198, 220, 215, 225, 190, 170, 16~
## $ weight       <int> 3504, 3693, 3436, 3433, 3449, 4341, 4354, 4312, 4425, 385~
## $ acceleration <dbl> 12.0, 11.5, 11.0, 12.0, 10.5, 10.0, 9.0, 8.5, 10.0, 8.5, ~
## $ year         <int> 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 70, 7~
## $ origin       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 3, ~
## $ name         <fct> chevrolet chevelle malibu, buick skylark 320, plymouth sa~
```

In the `Auto` data set, `mpg` is a quantitative variable but it is typically the response, not a predictor. The quantitative predictors are therefore `cylinders`, `displacement`, `horsepower`, `weight`, `acceleration`, and `year`. The qualitative predictors are `origin` (a categorical variable encoded numerically) and potentially `name` (more on `name` in part (e)).

Problem 2 (b)

```r
# Select quantitative predictors.
Auto_quant_preds <- Auto %>%
    dplyr::select(cylinders, displacement, horsepower, weight, acceleration, year)


# Compute range for each quantitative predictor.
ranges <- sapply(Auto_quant_preds, range)
tibble::tibble(
  Predictor = colnames(ranges),
  Min       = ranges[1, ],
  Max       = ranges[2, ],
  Range     = Max - Min
```

```r
)
```

```
## # A tibble: 6 x 4
##   Predictor      Min    Max  Range
##   <chr>        <dbl>  <dbl>  <dbl>
## 1 cylinders        3    8      5
## 2 displacement    68  455    387
## 3 horsepower      46  230    184
## 4 weight        1613 5140   3527
## 5 acceleration     8   24.8   16.8
## 6 year            70   82     12
```

```r
rm(ranges)
```

Problem 2 (c)

```r
# Compute mean and standard deviation for each.
tibble::tibble(
  Predictor = names(Auto_quant_preds),
  Mean      = sapply(Auto_quant_preds, mean),
  SD        = sapply(Auto_quant_preds, sd)
)
```

```
## # A tibble: 6 x 3
##   Predictor       Mean     SD
##   <chr>          <dbl>  <dbl>
## 1 cylinders       5.47   1.71
## 2 displacement  194.   105.
## 3 horsepower    104.    38.5
## 4 weight       2978.   849.
## 5 acceleration   15.5    2.76
## 6 year           76.0    3.68
```

Problem 2 (d)

```r
# Remove 10th through 85th observations (inclusive).
Auto_quant_subset <- Auto_quant_preds[-c(10:85), ]

# Compute range, mean, and standard deviation for each predictor on the subset.
ranges_sub <- sapply(Auto_quant_subset, range)
tibble::tibble(
  Predictor = colnames(ranges_sub),
  Min       = ranges_sub[1, ],
  Max       = ranges_sub[2, ],
  Range     = Max - Min,
  Mean      = sapply(Auto_quant_subset, mean),
  SD        = sapply(Auto_quant_subset, sd)
)
```

```
## # A tibble: 6 x 6
##   Predictor       Min    Max  Range    Mean     SD
##   <chr>         <dbl>  <dbl>  <dbl>   <dbl>  <dbl>
## 1 cylinders       3      8      5      5.37   1.65
## 2 displacement   68    455    387    187.    99.7
## 3 horsepower     46    230    184    101.    35.7
## 4 weight       1649   4997   3348   2936.   811.
## 5 acceleration    8.5   24.8   16.3   15.7    2.69
## 6 year           70     82     12     77.1    3.11
```

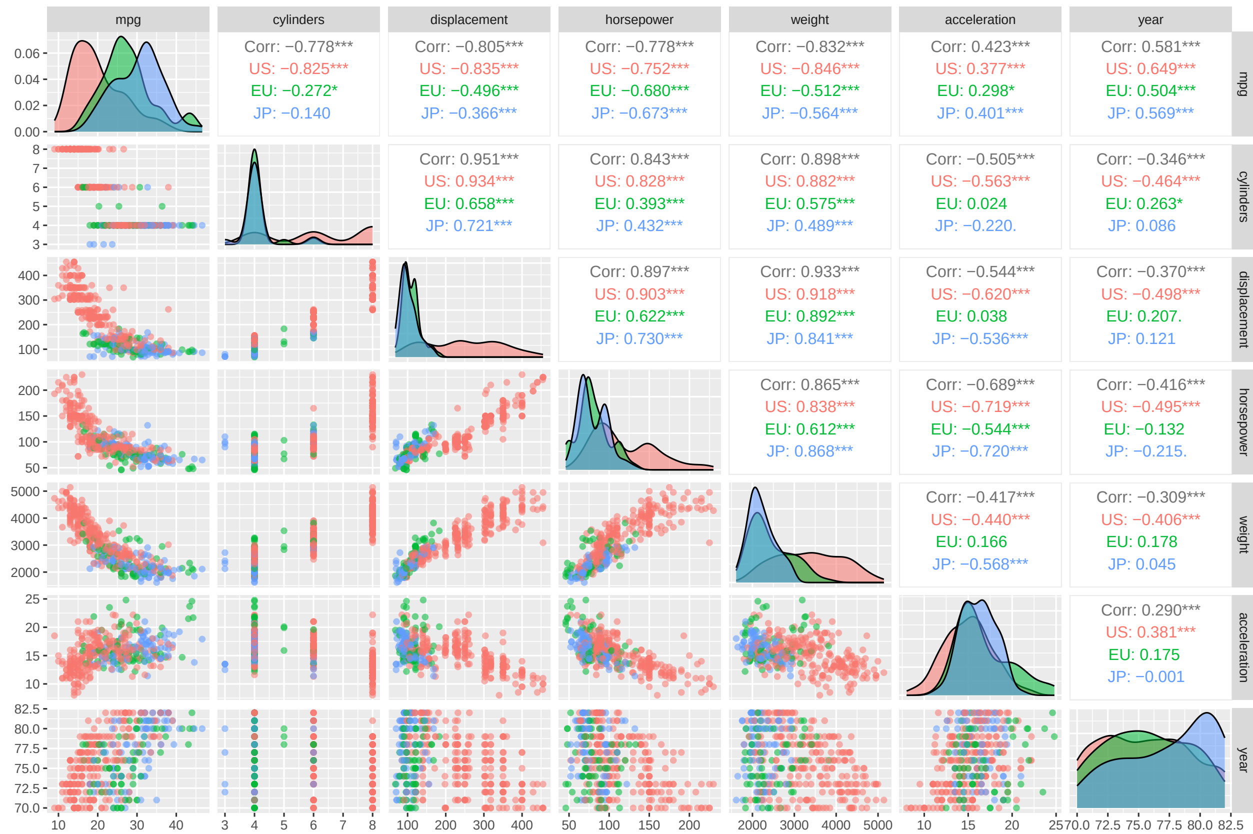
```r
rm(Auto_quant_preds, Auto_quant_subset, ranges_sub)
```

Problem 2 (e)

```r
# Factor origin.
Auto_plot <- Auto %>%
  dplyr::mutate(origin = factor(origin, labels = c('US', 'EU', 'JP')))

# Inspect pairwise relationships.
```

```r
GGally::ggpairs(
  Auto_plot,
  columns   = setdiff(colnames(Auto_plot), c('origin', 'name')),
  aes(color = origin, alpha = 0.5)
)
```




```r
rm(Auto_plot)

# Count number of unique names.
sum(table(Auto$name) == 1)
```

```
## [1] 245
```

The scatterplot matrix shows strong collinearity among the “size/power” predictors: `cylinders`, `displacement`, `horsepower`, and `weight` move together very tightly (e.g., `cylinders-displacement` has a correlation around 0.95, and `displacement-weight` around 0.93), suggesting these variables are largely measuring the same underlying concept (bigger engines/cars tend to be heavier and more powerful). In contrast, `acceleration` tends to be negatively associated with those size/power variables (most notably with `horsepower`, around -0.69, and with `displacement`, around -0.54), indicating that cars with larger engines and greater power/weight tend to have smaller `acceleration` values in this dataset. The variable `year` is moderately negatively related to the size/power measures (roughly -0.31 to -0.42 with `weight`, `displacement`, and `horsepower`) and mildly positively related to `acceleration` (about 0.29), consistent with cars becoming lighter and less "big-engine" over time. Finally, the color-group patterns by `origin` suggest systematic differences across regions (U.S. cars clustering at higher weight/displacement/horsepower), and the within-`origin` correlations sometimes differ (e.g., the `cylinders-acceleration` relationship is much stronger for U.S. cars than for European or Japanese cars), reinforcing that relationships among predictors can vary by subgroup even when the overall trend is clear. The `name` variable has an extremely large number of unique values (close to the number of observations), so it behaves mostly like an identifier rather than a reusable predictor. Treating it as a categorical feature would create a very high-cardinality factor with many rare levels, which tends to overfit and won't generalize well.

Problem 2 (f)

Yes. The plots suggest that several variables should be useful for predicting `mpg`. In particular, `mpg` has strong negative associations with `weight`, `displacement`, `horsepower`, and `cylinders` (heavier, larger-engine, higher-power cars tend to have lower gas mileage), so these predictors should have substantial predictive value. The plots also show that `mpg` increases with `year`, indicating that newer model years are generally more fuel-efficient, and `mpg` has a more moderate positive relationship with `acceleration`. Finally, the clear separation by `origin` in the panels suggests that `origin` is also informative: cars from different regions cluster at different typical fuel-efficiency levels even after accounting for other predictors, so it may help explain additional variation in `mpg` beyond the purely quantitative variables.

Problem 3 (a)

```r
# Load data.
data(Boston)

# Learn data.
# ?Boston

# Check dimensions.
dim(Boston)
```

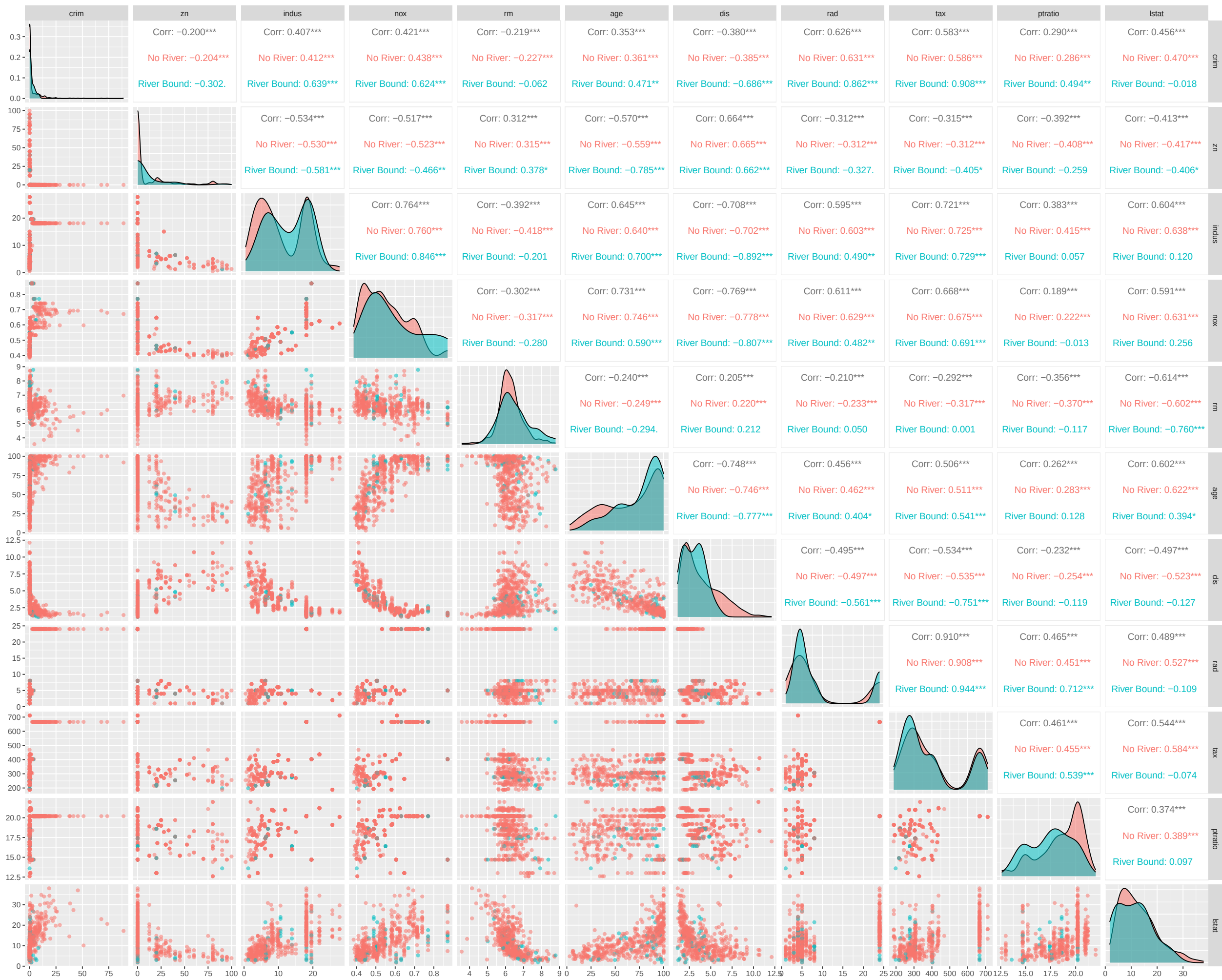
```
## [1] 506  13
```

There are 506 rows and 13 columns in the `Boston` data set. Each row corresponds to a census tract/suburb (help file says suburb; textbook says census tract) in the Boston area, and each column is a variable recorded for that tract. One of the columns is `medv`, the median home value (in $1000s), and the remaining columns are predictors describing characteristics of the tract (e.g., crime rate, number of rooms, property tax rate, etc.).

Problem 3 (b)

```
# Factor origin.
Boston_plot <- Boston %>%
  dplyr::mutate(chas = factor(chas, levels = c(0, 1), labels = c('No River', 'River Bound')))

# Inspect pairwise relationships.
GGally::ggpairs(
  Boston_plot,
  columns = setdiff(colnames(Boston_plot), c('medv', 'chas')),
  aes(color = chas, alpha = 0.5)
)
```




```r
rm(Boston_plot)
```

Many predictors are highly skewed, especially `crim`, `zn`, `rad`, and `tax`, with most observations near small values and a handful of extreme outliers, so relationships are often driven by a small number of high-leverage tracts. There is clear collinearity among "urban/industrial" variables: `indus` is strongly positively associated with `nox`, while `dis` is strongly negatively associated with both `indus` and `nox`, suggesting that more industrial and higher-pollution tracts tend to lie closer to employment centers. The variable `age` also tends to be higher where `dis` is lower, consistent with older housing stock being concentrated nearer the city. In addition, `rad` and `tax` are very strongly positively related, indicating that tracts with greater accessibility to radial highways tend to have higher property-tax rates. Finally, `lstat` is positively related to several "urban stress" measures (e.g., `crim`, `nox`, `tax`) and negatively related to variables associated with more desirable suburban tracts (e.g., `zn`, `dis`), reinforcing that multiple predictors are capturing overlapping aspects of neighborhood socioeconomic status and urbanization. The binary predictor `chas` (river adjacency) naturally appears as two bands; any differences involving `chas` are best interpreted as group-level shifts rather than a continuous linear trend.

Problem 3 (c)

Because `crim` is extremely right-skewed, the scatterplots look compressed and many relationships appear weak visually. Still, the correlation panel indicates moderate associations: `crim` is most positively related to `rad` and `tax` (and also `lstat`, `nox`, and `indus`), and it is moderately negatively related to `dis`. Other predictors (e.g., `zn`, `rm`) show weaker negative associations. Overall, the evidence suggests some predictors are associated with crime rate, but the relationships are not uniformly strong and are influenced by skew/outliers.

Problem 3 (d)

```r
# Count top 5 values of variable.
top5_value_counts <- function(df, var) {
  df %>%
    count(Value = .data[[var]], name = 'num_tracts') %>%  # frequency table
    arrange(desc(Value)) %>%                              # highest values first
    slice(1:5) %>%                                        # top 5 distinct values
    mutate(Variable = var, .before = 1)
}
top5_value_counts(Boston, 'crim')
```

```
##   Variable   Value num_tracts
## 1     crim 88.9762          1
## 2     crim 73.5341          1
## 3     crim 67.9208          1
## 4     crim 51.1358          1
## 5     crim 45.7461          1
```

```r
top5_value_counts(Boston, 'tax')
```

```
##   Variable Value num_tracts
## 1      tax   711          5
## 2      tax   666        132
## 3      tax   469          1
## 4      tax   437         15
```

```
## 5      tax   432          9
```

```r
top5_value_counts(Boston, 'ptratio')
```

```
##   Variable Value num_tracts
## 1  ptratio  22.0          2
## 2  ptratio  21.2         15
## 3  ptratio  21.1          1
## 4  ptratio  21.0         27
## 5  ptratio  20.9         11
```

```r
# Compute ranges.
ranges <- sapply(Boston %>% dplyr::select(crim, tax, ptratio), range)
tibble::tibble(
  Variable = colnames(ranges),
  Min      = ranges[1, ],
  Max      = ranges[2, ],
  Range    = Max - Min
)
```

```
## # A tibble: 3 x 4
##   Variable       Min   Max Range
##   <chr>        <dbl> <dbl> <dbl>
## 1 crim       0.00632  89.0  89.0
## 2 tax      187       711   524
## 3 ptratio   12.6      22     9.4
```

```r
rm(ranges, top5_value_counts)
```

For `crim`, yes, there are a handful of tracts with extremely high crime rates. The five largest observed `crim` values are approximately 88.98, 73.53, 67.92, 51.14, and 45.75, and each of these occurs only once (so they are isolated extreme outliers). In contrast, the minimum `crim` is about 0.006, so the range is enormous (roughly 89), indicating that crime varies dramatically across tracts and is dominated by a few very high-crime locations. For `tax`, there are also tracts with particularly high values, but the pattern is different: the maximum `tax` rate is 711 and it occurs in 5 tracts, while the next-highest value 666 occurs in a very large number of tracts (132). This indicates that high `tax` rates are not just isolated outliers; there are sizeable clusters of tracts at very high `tax` levels. The overall range is still large (711 down to 187, range 524), reflecting substantial variation across towns/tracts. For `ptratio`, the highest pupil-teacher ratio is 22.0 (2 tracts), and other top values like 21.2 (15 tracts) and 21.0 (27 tracts) occur for many tracts, so "high" pupil–teacher ratios are fairly common compared to the extreme crime values. However, the spread is much smaller overall: `ptratio` ranges from 12.6 to 22.0 (range 9.4), which is narrow relative to `crim` and `tax`. Overall, `crim` and `tax` show much more dramatic variability (with `crim` having a few extreme outliers and `tax` having large high-value clusters), whereas `ptratio` varies less and its high values are shared by many tracts.

Problem 3 (e)

```r
# Count tract(s) bounding Charles River.
sum(Boston$chas == 1)
```

```
## [1] 35
```

Problem 3 (f)

```r
# Compute median pupil-teacher ratio.
median(Boston$ptratio, na.rm = TRUE)
```

```
## [1] 19.05
```

Problem 3 (g)

```r
# Identify tract(s) with lowest medv.
Boston_id <- Boston %>%
  dplyr::mutate(tract = dplyr::row_number())
min_medv_tracts <- Boston_id %>% dplyr::slice_min(medv, n = 1, with_ties = TRUE)
min_medv_tracts
```

```
##      crim zn indus chas   nox    rm age    dis rad tax ptratio lstat medv tract
## 1 38.3518  0  18.1    0 0.693 5.453 100 1.4896  24 666    20.2 30.59    5   399
## 2 67.9208  0  18.1    0 0.693 5.683 100 1.4254  24 666    20.2 22.98    5   406
```

```r
# Define helper to make comparison tables.
make_comp_tbl <- function(tid, Boston_id, pred_names, ranges) {
```

```r
    row <- Boston_id %>% dplyr::filter(tract == tid)
    v   <- as.numeric(row[1, pred_names])
    tibble::tibble(
        Predictor = pred_names,
        Value     = v,
        Min       = ranges[1, ],
        Max       = ranges[2, ],
        Range     = Max - Min
    )
}

# Compare values.
tract_ids  <- min_medv_tracts$tract
pred_names <- setdiff(names(Boston), 'medv')
ranges     <- sapply(Boston %>% dplyr::select(dplyr::all_of(pred_names)), range)
make_comp_tbl(399, Boston_id, pred_names, ranges)
```

```
## # A tibble: 12 x 5
##    Predictor   Value      Min    Max   Range
##    <chr>       <dbl>    <dbl>  <dbl>   <dbl>
##  1 crim       38.4    0.00632  89.0    89.0
##  2 zn          0      0       100     100
##  3 indus      18.1    0.46     27.7    27.3
##  4 chas        0      0         1       1
##  5 nox         0.693  0.385     0.871   0.486
##  6 rm          5.45   3.56      8.78    5.22
##  7 age       100      2.9     100      97.1
##  8 dis         1.49   1.13     12.1    11.0
##  9 rad        24      1        24      23
## 10 tax       666    187       711     524
## 11 ptratio    20.2   12.6      22       9.4
## 12 lstat      30.6    1.73     38.0    36.2
```

```r
make_comp_tbl(406, Boston_id, pred_names, ranges)
```

```
## # A tibble: 12 x 5
##    Predictor   Value      Min     Max   Range
##    <chr>       <dbl>    <dbl>   <dbl>   <dbl>
##  1 crim       67.9    0.00632  89.0    89.0
##  2 zn          0      0       100     100
##  3 indus      18.1    0.46     27.7    27.3
##  4 chas        0      0         1       1
##  5 nox         0.693  0.385     0.871   0.486
##  6 rm          5.68   3.56      8.78    5.22
##  7 age       100      2.9     100      97.1
##  8 dis         1.43   1.13     12.1    11.0
##  9 rad        24      1        24      23
## 10 tax       666    187       711     524
## 11 ptratio    20.2   12.6      22       9.4
## 12 lstat      23.0    1.73     38.0    36.2
```

```r
rm(min_medv_tracts, make_comp_tbl, tract_ids, pred_names, ranges)
```

The lowest median home value `medv` occurs in two census tracts, 399 and 406, both with `medv = 5` (in \$1000s). Both tracts share a very similar high-stress urban profile, which helps explain why their home values are so low. In both tracts, the housing stock is as old as it gets in this dataset (`age = 100`, the maximum), they are very close to the city/employment core (`dis   1.43-1.49`, near the minimum), and they sit in a heavily urban/industrial environment (`indus = 18.1` and `nox = 0.693`, both high). They also have the maximum highway accessibility (`rad = 24`, the dataset maximum) and very high property tax (`tax = 666`, near the top of the range), consistent with the strong `rad-tax` relationship seen elsewhere in the dataset. Where the two tracts differ most is the severity of crime and socioeconomic stress. Tract 399 has `crim = 38.35` together with an extremely high `lstat = 30.59` (near the upper end of the dataset), while tract 406 has even more extreme crime (`crim = 67.92`, among the very highest values) with still-elevated `lstat = 22.98`. In addition, both tracts have relatively low numbers of rooms per dwelling (`rm = 5.45` and `rm = 5.68`), suggesting smaller/less valuable housing compared with higher-`medv` areas.

Problem 3 (h)

```r
# Count tracts with rm > 7 and rm > 8.
tibble::tibble(
  Condition = c('rm > 7', 'rm > 8'),
  Count     = c(sum(Boston$rm > 7), sum(Boston$rm > 8))
)
```

```
## # A tibble: 2 x 2
##   Condition Count
##   <chr>     <int>
## 1 rm > 7       64
## 2 rm > 8       13
```

```r
# Inspects tracts with rm > 8.
Boston_id %>%
  dplyr::relocate(tract, .before = 1) %>%
  dplyr::filter(rm > 8) %>%
  dplyr::arrange(dplyr::desc(rm))
```

```
##    tract    crim zn indus chas    nox    rm  age    dis rad tax ptratio lstat
## 1    365 3.47428  0 18.10    1 0.7180 8.780 82.9 1.9047  24 666    20.2  5.29
## 2    226 0.52693  0  6.20    0 0.5040 8.725 83.0 2.8944   8 307    17.4  4.63
## 3    258 0.61154 20  3.97    0 0.6470 8.704 86.9 1.8010   5 264    13.0  5.12
## 4    263 0.52014 20  3.97    0 0.6470 8.398 91.5 2.2885   5 264    13.0  5.91
## 5    164 1.51902  0 19.58    1 0.6050 8.375 93.9 2.1620   5 403    14.7  3.32
## 6    233 0.57529  0  6.20    0 0.5070 8.337 73.3 3.8384   8 307    17.4  2.47
## 7    268 0.57834 20  3.97    0 0.5750 8.297 67.0 2.4216   5 264    13.0  7.44
## 8    225 0.31533  0  6.20    0 0.5040 8.266 78.3 2.8944   8 307    17.4  4.14
## 9    254 0.36894 22  5.86    0 0.4310 8.259  8.4 8.9067   7 330    19.1  3.54
## 10   234 0.33147  0  6.20    0 0.5070 8.247 70.4 3.6519   8 307    17.4  3.95
## 11    98 0.12083  0  2.89    0 0.4450 8.069 76.0 3.4952   2 276    18.0  4.21
## 12   227 0.38214  0  6.20    0 0.5040 8.040 86.5 3.2157   8 307    17.4  3.13
## 13   205 0.02009 95  2.68    0 0.4161 8.034 31.9 5.1180   4 224    14.7  2.88
##    medv
```

```
## 1  21.9
## 2  50.0
## 3  50.0
## 4  48.8
## 5  50.0
## 6  41.7
## 7  50.0
## 8  44.8
## 9  42.8
## 10 48.3
## 11 38.7
## 12 37.6
## 13 50.0
```

```
rm(Boston_id)
```

In this data set, 64 census tracts have an average of more than 7 rooms per dwelling (`rm > 7`), and 13 census tracts have an average of more than 8 rooms per dwelling (`rm > 8`). The `rm > 8` tracts are relatively rare and, based on the printed rows and contrasting with the roughly estimated ranges deduced from the pair plots in part (b), generally resemble higher-end neighborhoods: many have very high median home values (`medv` is frequently near the top of the scale, including several tracts with `medv = 50`), and their `lstat` values tend to be low (roughly 2–7), which is consistent with more affluent areas. Crime rates for these tracts are typically low to moderate (most `crim` values are well below the extreme outliers seen elsewhere in the dataset), though there is at least one clear exception: tract 365 has the largest `rm` value (`rm = 8.78`) but also relatively high `crim` and very high `nox`, showing that very large homes can coexist with some urban disamenities. Overall, tracts with `rm > 8` tend to correspond to especially desirable areas with high home values and favorable socioeconomic indicators, with a small number of exceptions.