Homework 2 - Regression

Instructions

Due 18 October at 11pm

Instructions

- Use the space inside of

```
::: {.solbox data-latex=""}
```

```
:::
```

to answer the following questions.

- Do not move this file or copy it and work elsewhere. Work in the same directory.
- Use a new branch named whatever you want. Create it now! Can't come up with something, try here. Make a small change, say by adding your name. Commit and push now!
- Try to Knit your file now. Are there errors? Fix them now, not at 11pm on the due date.
- There MUST be some text between `::: {.solbox}` and the next `:::` or this will fail to Knit.
- If your code or figures run off the .pdf, you'll lose 2 points automatically.

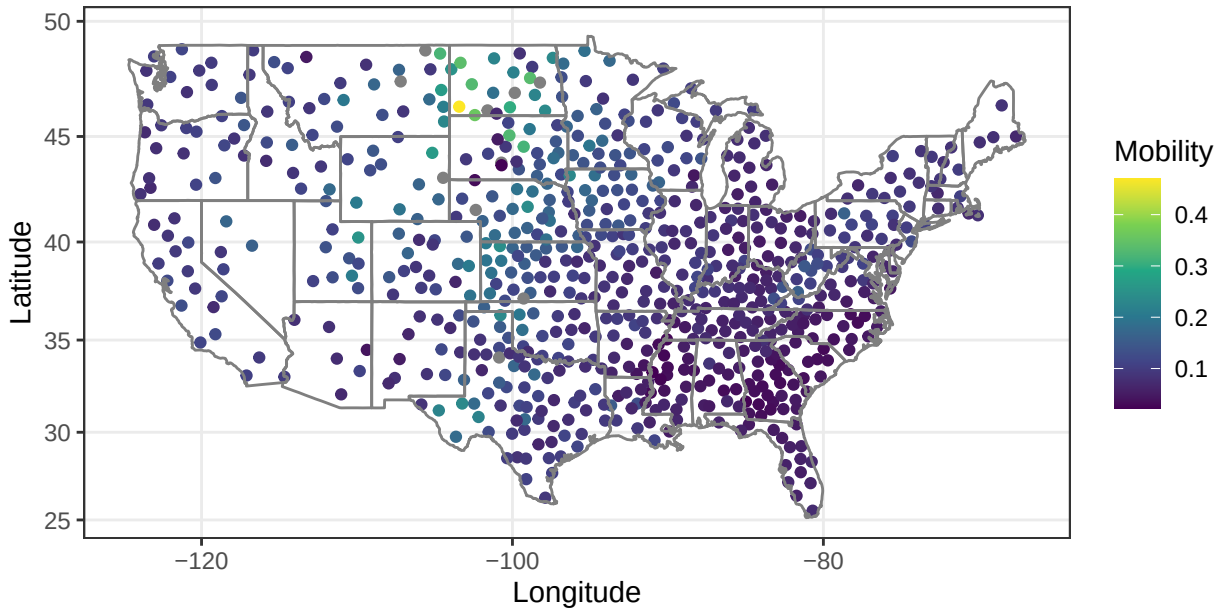
Regularized methods

In this section we will conduct an analysis of the `mobility` data first visited during the first week of class. There is a file `mobility.html` in your repo which gives descriptions of all the variables. Load it in a web browser to find out about the covariates available to you.

This assignment will look at economic mobility across generations in the contemporary USA. The data come from a large study, based on tax records, which allowed researchers to link the income of adults to the income of their parents several decades previously. For privacy reasons, we don't have that individual-level data, but we do have aggregate statistics about economic mobility for several hundred communities, containing most of the American population, and covariate information about those communities. We are interested in predicting economic mobility from the characteristics of communities. Note that some observations are missing values for some covariates.

1. The following code generates a map of `Mobility` ignoring Alaska and Hawaii. Describe the geographic pattern in words.

```
ggplot(
  mobility %>% filter(!(State %in% c("AK", "HI") )),
  aes(x=Longitude, y=Latitude, color = Mobility)) +
  geom_point() +
  coord_map() +
  borders("state") +
  scale_color_viridis_c()
```



(placeholder text)

2. Make scatter plots of mobility against each of the following variables (by "plot mobility against B" we mean "put mobility on the y-axis and B on the x-axis"): `Population`, `Income`, `Seg_racial`, `Share01`, `School_spending`, `Violent_crime`, and `Commute`. Include on each plot a line for the univariate regression of mobility on the variable, and give a table of the slope coefficients. Carefully explain the interpretation of each coefficient in the context of the problem. You likely need to look at the documentation to determine what these variables mean. **Hint:** This can be done easily with `ggplot() + geom_smooth()`. See the Examples here. If you choose a different route, I suggest writing a function.

```
#some code
```

(placeholder text)

3. Run a linear regression of `mobility` against all appropriate covariates.
    a. Report all regression coefficients and their standard errors to reasonable precision; you may use either a table or a figure as you prefer. Do not just paste in R's output.
    b. Explain why the `ID` variable must be excluded.
    c. Explain which other variables, if any, you excluded from the regression, and why. (If you think they can all be used, explain why.)
    d. Compare the coefficients you found in problem 2 to the coefficients for the same variables in this regression. Are they much different? Have any changed sign?

```
ols <- lm(Mobility ~ . , data = mobility) # YOU NEED TO CHANGE THIS LINE!!

# some code
```

a.
b.
c.
d.

4. With all the covariates you used in the previous question, use ridge regression and lasso (with the `{glmnet}` package). Use cross validation as implemented in `cv.glmnet()`.
    a. Plot the CV curve for Lasso. Explain the difference between the two vertical lines shown on the figure. What are the numbers on the top of the plot?
    b. Plot the coefficient trace for ridge regression. What does "L1 norm" on the x-axis mean? Are any coefficient estimates exactly 0 for any value of the penalty parameter? As $\lambda \to 0$, which way do I move on the x-axis?

a.

```
library(glmnet)
x <- model.matrix(ols) # grabs the design matrix from the internal lm() output
x <- x[, -1] # remove the intercept column, glmnet() will do this for us
y <- mobility$Mobility[-ols$na.action] # remove those observations with
# missing values. glmnet() hates those. They are already dropped in x.
set.seed(01101101)
```

b.

```
# more code
```

5. For both the Ridge regression and Lasso regression in the previous section, choose the set of coefficient estimates using `lambda.min`. Compare these coefficient estimates with those in problem 3 by producing a graph (put the Variable names on the y-axis, the coefficient estimate on the x-axis, and use color or shape to denote the three methods). Describe what you see.

```
# some code
```

(placeholder text)

6. Calculate the LOOCV score for your OLS model in problem 3. Compare this score with the scores from problem 4. Select 1 of these three models to use for out-of-sample prediction. Explain why you chose this model. For your chosen model, make a map of the residuals (borrow code from problem 1). Describe any patterns you see. But change the coloring so that 0 is white (use `scale_color_gradient2()`)

```
set.seed(123456)
# some code
```

(placeholder text)

```
# perhaps more code
```

(placeholder text)

Nonparametric regression

It may be helpful to read through the Kernel Regression section of the worksheets before starting this section.

```
data("bccovid")
```

1. Plot the data on the log-scale along with a 7-day trailing average: $\overline{y}_i = \sum_{j=i-6}^{i} y_j$. This is one of the most common ways to "smooth" this data (see e.g. CBC). If you install the `R` package `{zoo}`, there's a function to do this easily with appropriately chosen arguments (though you can write your own!). How well does this smoother "track" the data? Do you notice any discrepancies? Issues? It may be helpful to look at the most recent 2 months of data to answer these questions.

```
# some code
```

(placeholder text)

2. The 7-day trailing average is a "linear smoother". Complete the function below that creates the smoothing matrix for any dimension `n` and any "window" of days `k`. You may assume that the data is equally spaced and arranged in increasing order. Think carefully about how to handle the first few rows. Your resulting matrix must be square. Evaluate your function for `n = 8`, and `k = 3` (round the entries to 2 decimals so it prints nicely).

```
trail_mean_mat <- function(n, k = 3) {
  stopifnot(n == floor(n), k == floor(k), n > 0, k > 0) # check for valid inputs
  mat <- matrix(0, nrow = n, ncol = n) # preallocate space
  for (i in 1:n) { # loop over rows
    idx <-   # which columns are nonzero?
    denom <- length(idx)
    mat[i, idx] <- 1 / denom
  }
  mat
}
round(trail_mean_mat(8, 3), 2)

## Error in trail_mean_mat(8, 3): object 'idx' not found
```

3. What is the effective degrees of freedom of the "trailing average" for $n = 50$ and $k = 10$? For n large relative to k, what does this look like for arbitrary k? (I'm being a bit hand-wavy here. You can examine $lim_{n\to\infty} edf(n, k)/n$ and convert to get the answer in terms of n and k.)

(placeholder text)

4. The following function generates the smoothing matrices and returns fitted nonparametric regression estimates and the EDF for Gaussian and Boxcar kernels with different bandwidths.

```
kernel_smoother <- function(x, y, kern = c("boxcar", "gaussian"), band = 7) {
  dmat <- as.matrix(dist(x))
  kern <- match.arg(kern)
  W <- switch(kern,
              boxcar = dmat <= (band * 0.5),
              gaussian = dnorm(dmat, 0, sd = band * 0.3706506))
  W <- sweep(W, 1, rowSums(W), '/')
  fit <- W %*% y
  out <- list(fit = fit, edf = sum(diag(W)))
  out
}
```

Use just the data on or after 1 July 2022. Use the function to plot the data (as points) along with (as lines) (1) the trailing 7-day average, (2) the boxcar smoother with `band = 7` and (3) the Gaussian smoother with `band = 7`. How do the results from the two new smoothers compare with those of the 7-day trailing average? Explain.

```
# some code
```

(placeholder text)

5. Adjust the `kernel_smoother()` function so that it also computes and returns the LOO-CV score. Compute the the LOO-CV score for each integer value of `band` from 1 to 21 for the Gaussian kernel. Plot the scores against `band`. Which value is best? Will the resulting plot be smoother or wigglier than with `band = 7`? Can you think of any reasons that this is best bandwidth by this metric? Should we use it?

```
kernel_smoother <- function(x, y, kern = c("boxcar", "gaussian"), band = 7) {
  # copy the code from above and add a few lines

}



bws <- 1:21
## some code
```

Some explanation.