HW 2: OLS and Linear Algebra[*]

due 6/21/2023

Introduction

Getting Started

Please note that this assignment pre-supposes that you have completed the steps in the GitHub Handout!

Each of your assignments will begin with the following steps. You saw these once already during the GitHub Handout activity, but they're outlined briefly below as a reminder. Refer back to first homework assignment or the GitHub Handout for a detailed list of the steps involved for getting started with an assignment.

- Click on the assignment link on Canvas. You will then be asked to accept the assignment.
- Once you accept the assignment, GitHub will create the repo for you. You will have to refresh the page for the link to the repo to show up.
- Click on the link - this will take you to your repo. This repo contains a template that you will build on to complete your assignment.
- On GitHub, click on the green **<> Code** button, select **HTTPS** (this might already be selected by default). Click on the clipboard icon (overlapping squares) to copy the repo URL.
- Now go back to Posit Cloud, making sure that you are in the Applied Machine Learning - Summer 2023 workspace.
- Then, click on "New Project" near the top right-hand corner of the page. This will open a drop-down selection list. Here, you will want to click on "New Project from Git Repository".
- Paste the URL you copied from GitHub into the dialogue box and press "OK"

[*]Adapted from homework from Data Science in a Box, inspired by materials from ISLR, 2nd Edition.

Overview

This homework is a chance to review and apply what you have learned about OLS in your readings and to practice data visualization a bit more. It also serves to illustrate the importance of linear algebra in the context of machine learning.

Topic: Bike Sharing

Figure 1: Photo by Viktor Kern on Unsplash

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return back has become automatic. Through these systems, user is able to easily rent a bike from a particular position and return back at another position.

Currently, there are over 500 bike-sharing programs around the world, comprising over 500 thousands bicycles. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues.

Apart from interesting real world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be detected via monitoring these data.

Source: UCI Machine Learning Repository - Bike Sharing Dataset

Warm Up

Before you get started with the assignment, let's warm up by adding your name and date to your document. Update the YAML of `hw2.qmd` to do so.

Once you have done so, render the document, stage the `.qmd` and `.pdf` files, commit them, and then push to your repo on GitHub. Make sure that the changes show up there.

! File Name

You do not need to change the name of your `.qmd` file. It can stay `hw2.qmd`. If you change the file extension in some way by accident, you may run into rendering problems. Please just make sure to put the correct name (i.e. your name) on your document by updating the YAML!

Packages

We'll use the **tidyverse** package for much of the data wrangling and visualisation. We will be using base R for fitting linear models, so we don't need to load any special packages for that.

Make sure to load them in a chunk in your `hw2.qmd` file.

```
#loading packages
```

Data

The data can be found in the "data/bikeshare-day.csv" file. Import it as `dcbikeshare`. **Note**: file paths inside `.qmd` files are *relative* by default – you don't need to specify a full (i.e. *absolute* file path), you can use a file path *relative* to the location of the `.qmd` file.

For example, if your `hw2.qmd` file is located in a directory with absolute file path `C:/some/file/path/hw2-hoellers`, and the data file is at `C:/some/file/path/hw2-hoellers/data/bikeshar` then the `data` folder and your `hw2.qmd` file are located in the same folder. RStudio knows where the `.qmd` file is. To load it, you do not need to use `read_csv("C:/some/file/path/hw2-hoellers/data/bikesh` you can just use `read_csv(data/bikeshare-day.csv")`.

```
#import data here
```

The codebook is below:

Variable name	Description
`instant`	record index
`dteday`	date
`season`	season (1:winter, 2:spring, 3:summer, 4:fall)
`yr`	year (0: 2011, 1:2012)
`mnth`	month (1 to 12)
`holiday`	whether day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)
`weekday`	day of the week
`workingday`	if day is neither weekend nor holiday is 1, otherwise is 0.
`weathersit`	1: Clear, Few clouds, Partly cloudy, Partly cloudy 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
`temp`	Normalized temperature in Celsius. The values are divided by 41 (max)
`atemp`	Normalized feeling temperature in Celsius. The values are divided by 50 (max)
`hum`	Normalized humidity. The values are divided by 100 (max)
`windspeed`	Normalized wind speed. The values are divided by 67 (max)
`casual`	Count of casual users
`registered`	Count of registered users
`cnt`	Count of total rental bikes including both casual and registered

The data include daily bike rental counts (by members and casual users) of Capital Bikeshare in Washington, DC in 2011 and 2012 as well as weather information on these days. The original data sources are http://capitalbikeshare.com/system-data and http://www.freemeteo.com.

Exercises

Data wrangling

Note: For these questions, pay special attention to the data wrangling slides. Remember that you can change/update variables using the `mutate()` function. Also, the `fct_relevel()` and `case_when()` functions will be helpful here.

Q1

Recode the `season` variable to be a factor with meaningful level names as outlined in the codebook (at the link above), with "spring" as the baseline level.

Q2

Recode the binary variables `holiday` and `workingday` to be factors with levels "no" (0) and "yes" (1), with "no" as the baseline level.

Q3

Recode the `yr` variable to be a factor with levels 2011 and 2012, with 2011 as the baseline level.

Q4

Recode the `weathersit` variable as 1 - "clear", 2 - "mist", 3 - "light precipitation," and 4 - "heavy precipitation", with "clear" as the baseline.

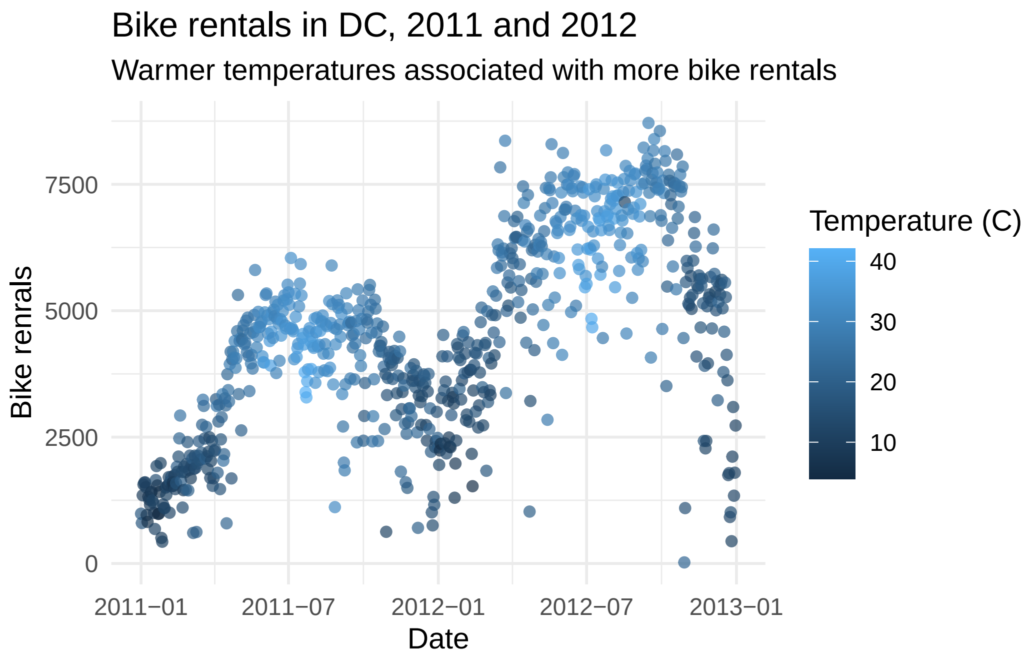
Q5

Calculate raw temperature, feeling temperature, humidity, and windspeed as their values given in the dataset multiplied by the maximum raw values stated in the codebook for each variable. **Instead of writing over the existing variables, create new ones with concise but informative names.**

Exploratory data analysis

Q6

Recreate the following visualization, and interpret it in context of the data. **Hint:** You will need to use one of the variables you created above. The temperature plotted is the feeling temperature.



Modeling

For this section, use the *raw* versions of the weather variables (temperature, humidity, etc)

Q7

Use `lm()` to fit a simple linear regression model predicting total daily bike rentals from daily temperature. Use the `summary()` function to print the results. Comment on the output:

(a) Is there a relationship between the predictor and the response?
(b) How strong does this relationship seem to be?
(c) Is the relationship between the predictor and the response positive or negative?
(d) What is the predicted `cnt` associated with a temperature of 63 degrees Fahrenheit? What are the associated 95% confidence and prediction intervals?

Q8

Fit another linear model predicting total daily bike rentals using daily feeling temperature. Use the `summary()` function to print the results. Is temperature or feeling temperature a better predictor of bike rentals? Explain your reasoning.

Q9

Fit a model predicting total daily bike rentals from season, year, whether the day is holiday or not, whether the day is a workingday or not, the weather category, temperature, feeling temperature, humidity, and windspeed, as well as the interaction between feeling temperature and holiday. Use the `summary()` function to print the results. Comment on the output:

(a) Is there a relationship between the predictors and the response?
(b) Which predictors appear to have a statistically significant relationship to the response?
(c) What do the coefficients for the `season` variable suggest?

Using Linear Algebra to Do Regression

As the textbook makes clear in Chapter 1 and as we will see as we move further though it, linear algebra – and in particular matrix operations – is at the core of many machine learning algorithms.

We haven't formally talked about how OLS fits into machine learning just yet (that's coming soon!), but it turns out that OLS is a rather simple approach to machine learning. In this section, you will see how OLS itself relies on linear algebra to get coefficient estimates.

Page 74 of ISLR explains that "multiple regression coefficient estimates have somewhat complicated forms that are most easily represented using matrix algebra." It turns out that we can minimize RSS (equation 3.22 in the book) with respect to the β, that is, the *vector* of coefficients, using matrix calculus. We won't go into the details of that here (we recommend that you take a statistics course on the linear model if you are interested) but we do want to demonstrate the connection!

Q10

Create a matrix called `X` that contains the version of the data used to fit the model above. We want `X` to contain only numbers – no factors or characters! So we will have to one-hot encode (use contrasts for qualitative predictors, in the language of the book) the non-numeric variables. The easiest way to do that is actually to use the `model.matrix()` function on the model fit object you created in Q9.

For example:

```r
model_object <- lm(response ~ predictor1 + predictor2)
model.fit(model_object)
```

The `model.matrix()` function is extremely useful – we will use it to get data ready for other machine learning methods in the future.

Q11

Create a vector called `y` that contains only the outcome variable for our regression. Because `model.matrix()` only creates the *design* matrix (the predictors), we need to use the `model.frame()` function here. This creates a data frame that contains *only* the rows used to actually fit the model. Why is this important? Because OLS (like a good number of machine learning approaches) cannot deal with NAs!

Q12

Now, use matrix multiplication to compute $(X^\top X)^{-1}X^\top y$. Print out the resulting vector (for self-comprehension: why is the output a vector?).

Hints:

- $^\top$ means *transpose – when we flip the rows and columns of a matrix. We can use the `t()` function to take the transpose of a matrix in R.
- $(A)^{-1}$ is the inverse-matrix of A. Matrix-inversion is a very important linear algebra operation. Inverse matrices are important because $AA^{-1} = I$ – a matrix times its inverse equals the identity matrix, one with 1s along the diagonal and 0s everywhere else. We can use the `solve()` function to compute the inverse of a matrix in R.

Compare $(X^\top X)^{-1}X^\top y$ to what `lm()` produced in question 11. What do you notice?

Grading

Remember to put your name and the date on your homework. You will *lose* points if this is not the case.

Remember to comment your code. You don't get points for commenting, but you will *lose* points for not commenting your code.

Remember to suppress warnings and messages. You will *lose* points if you do not suppress warnings and messages.

Remember to always directly answer any questions asked using a written response.

Remember to label all code chunks!

All plots should follow best visualization practices, including an informative title, labeled axes, and careful consideration of aesthetic choices. You will *lose* points if your plots are unmodified.

- Q1: 1 pt
- Q2: 1 pt
- Q3: 1 pt
- Q4: 1 pt
- Q5: 1 pt
- Q6: 5 pts (3 for plot, 2 for interpretation)
- Q7: 5 pts (2 pts for fitting model and returning summary, 3 for interpretation)
- Q8: 3 pts (2 pts for fitting model and returning summary, 1 pt for answering question and explaining reasoning)
- Q9: 6 pts (2 pts for fitting model and returning summary, 4 for interpretation)
- Q10: 1 pt
- Q11: 1 pt
- Q12: 3 pts
- Workflow (includes making sufficient commits and labeling chunks): 4 pts

Overall: 33 pts