ADS2 Problem Set 25: Preparing data for machine learning

ADS2

Semester 2, 2020/21

This problem set is for further exploration, with ideas on what else to do with the "handwritten digits" data. We have no estimate of the approximate time it will take, because that depends on how much you want to do. Spend as much time or as little time as you want.

Please make use of the Slack channel to share your ideas and get help.

Review of this week's practical

This week's practical was inspired by the following blog post by David Robinson (2018): Exploring handwritten digit classification: a tidy analysis of the MNIST dataset. http://varianceexplained.org/r/digit-eda/

You will see that in practical, we did things a bit more slowly. For instance, we went slowly through the steps of gathering the pixels that belong to each specific sample into rows and columns. In the blog entry, it is done in one single command that makes repeated use of the %>% operator. This is called the "pipe operator". It means "Take the output of the previous command and use it as input for the next". This makes coding more efficient if you know what you are doing, but it also makes it a bit more difficult for outsiders to understand a piece of code.

Going through the command to create pixels_gathered, see if you understand what it is doing, and how it is similar to what you did in Practical.

Further exploration

The same blog post has other ideas of what can be done with this data set, such as, for instance, plotting the brightness of all pixels on a histogram, to understand how it is distributed. It also looks at "atypical" digits, and at comparing two digits. Follow along some or all of it to see how you get on.

When you see a bit of code you don't understand there are several ways to deal with it.

- You could try and understand (either from the code itself or from the explanatory text) what the code is trying to do, and implement your own version of it, based on methods you are familiar with (similar to what we did in practical)
- You can copy and paste the code into your R script and see if it works there. If not, find out why not. If it does work, find out what things you can change to make it work differently. Understand the code by playing with it.
- (Of course you could also just copy and paste the code, run it, and be done in a short time. But you wouldn't really learn a lot.)