Fundamentals of Statistical Learning

Due at any time on Moodle before February 21st

THE Homework

Introduction

In this homework you will mainly work on the following main paper in order to implement a *conformal prediction* procedure aimed at *functional data*.

Conservation Note: be aware that the most difficult part of this exercise is to skim through the useless technicalities of the paper and squeeze it to the bare practical minimum relevant for you.

So, before digging any deeper into this homework, you should...

1. *Read the bare minimum on Functional Data Analysis (`FDA`)*
   In essence, in `FDA` a single datapoint, a single observation is not just a number or a label as usual, but a more complex beast, usually a whole function that we observe sampled at a given frequency. To deepen your understanding you may start, for example, from the first and second chapter of Ramsay and Silverman's book.
2. *Read the bare minimum on Conformal Prediction (`CP`)*
   In a nutshell, when doing estimation, we usually provide *confidence intervals* in addition to point estimates. Is there a similar notion for predictions? The answer is yes: we provide prediction *sets* or *set-valued* predictions, and `CP` is yet another (not so new) model agnostic, distribution-free technique designed for this purpose. This is a pretty hot topic at the moment, and you'll find tons of references out there (example). The only problem with most of these review works is their focus on *supervised* learning tasks $\rightsquigarrow$ **not** our case, we are working essentially *unsupervised*. So, while the ideas and the algorithms are very similar, try to keep it in focus and start from the notes and possibly Section 2 of the main paper.

$\rightsquigarrow$ Your job $\leftsquigarrow$

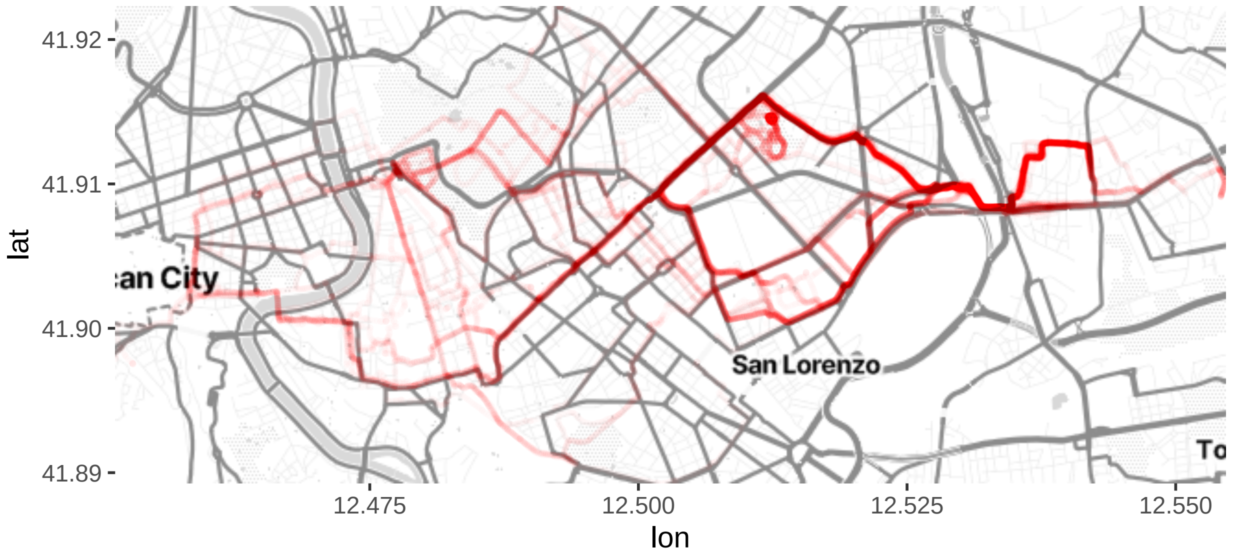
1. **Build or select a suitable dataset**
   - [**Aiming at a full grade**]
     Build your **own** "functional" dataset. In a separate doc, describe in details the data collection phase, possibly with pictures or...short movies, if useful.
   - [**Aiming at a grade** $\leqslant 26$]
     Select a publicly available dataset with some intrinsic interest. Explain your choice.
2. **Describe your data**
   - These are functional data $\rightsquigarrow$ you may use classical statitics and plots, but also dedicated ones (see, for example, our ref). Comment your results.
3. **Conformal step**
   - In *Section 3* and *Section 4* of the main paper the Authors propose two different approches to build conformal predictions for functional data. Pick one of the two, make your own `R` implementation, and then apply it on your data properly commenting 1. your data handling, 2. the results.

On collecting your *own* data

Once you've read a bit about functional data, you may come up with different ideas to collect your own dataset in a relatively short amount of time. Go ahead, roll with it. Nevertheless, if you feel a bit stuck, here are few options for you.

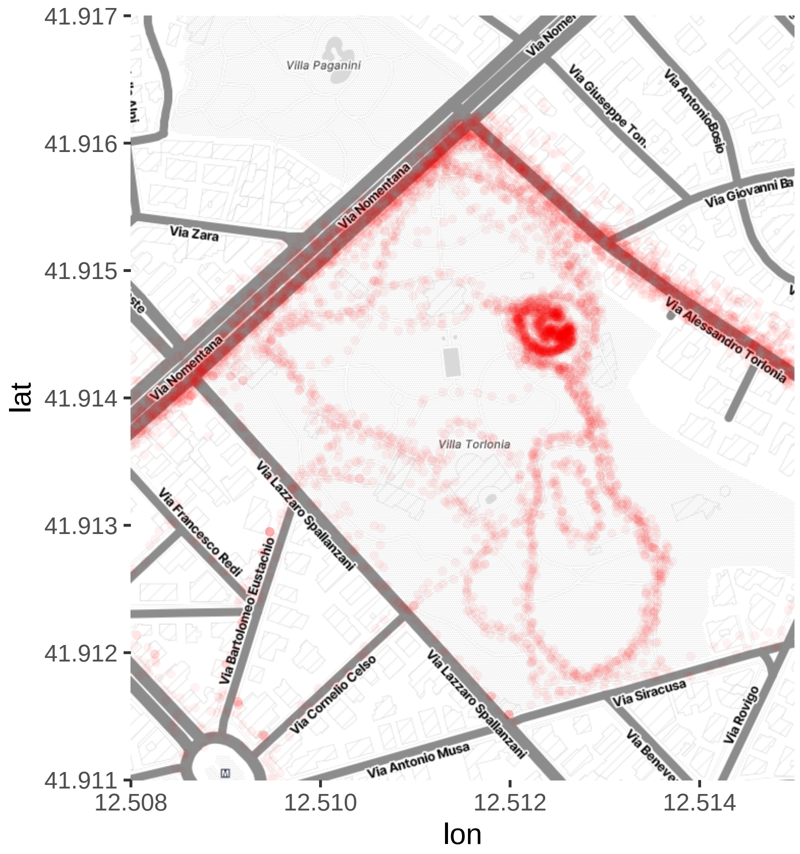
First of all, to leverage the sensors inside your phone, over the last couple of years I suggested Science Journal as a tool (now called Arduino Science Journal, see also here). You may or may not use it (for a variety of reasons), no problem.

But personal data coming from (personal) tracking devices are massive these days. So, let me take a very… *personal*… point of view on the issue. Over the years I build a decently sized dataset that collects the `gps` info of my (old) running sessions. To take a look we can use the `ggmap` package as follow:



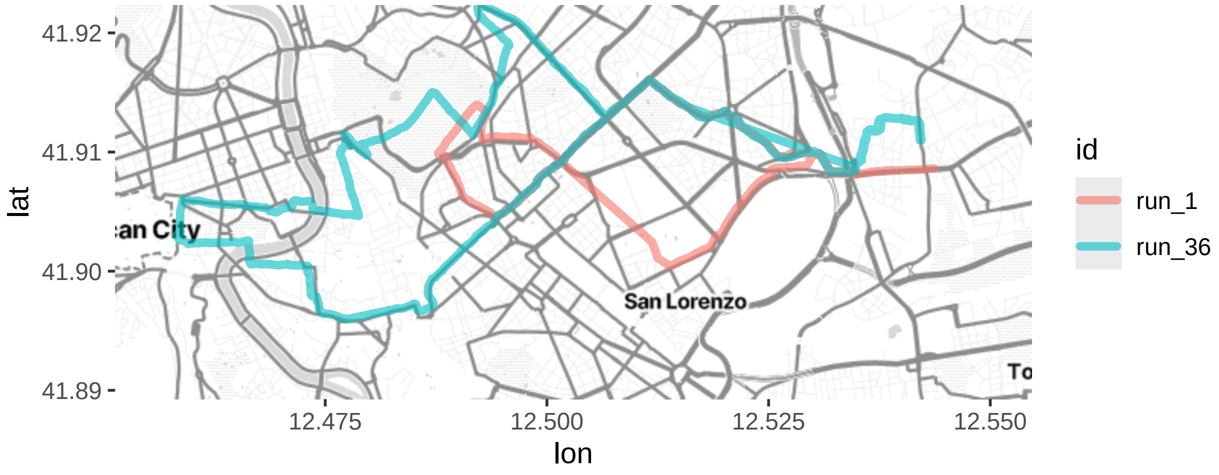
```
# The package
# url: https://github.com/dkahle/ggmap
require(ggmap)

# Get the map from Stadia
# To retrieve the API (no credit card required), go here:
# url for API: https://client.stadiamaps.com/signup
# register_stadiamaps(key = "PLACE YOUR STADIA API HERE")

# The data
load("trackme.RData") # not shared, it's here only for reference

# Map boundaries
myLocation <- c(min(runtrack$lon, na.rm = T),
                min(runtrack$lat, na.rm = T),
                max(runtrack$lon, na.rm = T),
                max(runtrack$lat, na.rm = T))

# Get the map from Stadia
myMapInD <- get_map(location = myLocation, source = "stadia",
                    maptype = "stamen_toner_lite", zoom = 13)

# Plot gps coordinates (without elevation data)
gp <- ggmap(myMapInD) + geom_point(data = runtrack,
                                    aes(x = lon, y = lat),
                                    size = .5, colour = I("red"), alpha = .01)
# Take a look
print(gp)
```

```
# Zoom in / Restricted map boundaries
reLocation <- c(12.508, 41.911, 12.515, 41.917)
# Get the map from Google (default) and plot
reMapInD <- get_map(location = reLocation, source = "stadia",
                    maptype = "stamen_toner_lite")
ggmap(reMapInD) + geom_point(data = runtrack, aes(x = lon, y = lat),
                             size = 1, colour = I("red"), alpha = .05)
```



Those red dots are not really dots, they are part of a single track, a single curve in the plane. Let's look at two of them:

```r
# Plot gps coordinates (without elevation data)
runsmall <- subset(runtrack, id %in% c("run_1", "run_36"))
gp2 <- ggmap(myMapInD) + geom_path(data = runsmall,
                                   aes(x = lon, y = lat, col = id),
                                   size = 1.5, lineend = "round",
                                   alpha = .6)
# Take a look
print(gp2)
```



Of course we are in the realm of *functional data analysis*: a single datapoint here is the two-dimensional curve describing the route associated with my running session (as sampled by my `gps` watch).