ETC 2420/5242 Lab 8 2016

Di Cook

Week 8

Purpose

In this lab we will pull data together from multiple sources to answer the question whether adverse weather affects pedestrian traffic in Melbourne.

Data

- Melbourne Open data portal contains pedestrian counts by hour for numerous sites around the city.
- The US Department of Commerce hosts a global data base on weather from ground stations collated by National Climatic Data Center.

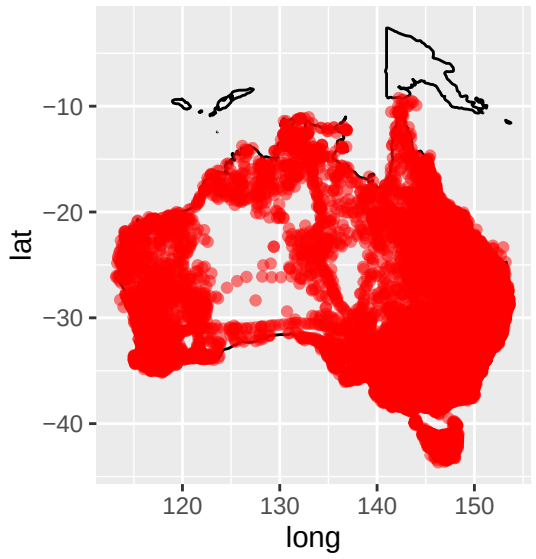
Getting started

The list of weather stations is a small file. We will pull this data from the web first. And plot the locations for Australia on a map. Data can be found at this web site http://www1.ncdc.noaa.gov/pub/data/ghcn/daily/. Download the file `ghcnd-stations.txt`.

The `read_table` function from the `readr` package uses whitespace to guess where records start and end, very convenient. We also will only read the station information for Australia, that have ids starting with `ASN`. I used a text editor to find the line numbers of the start and end of these records.

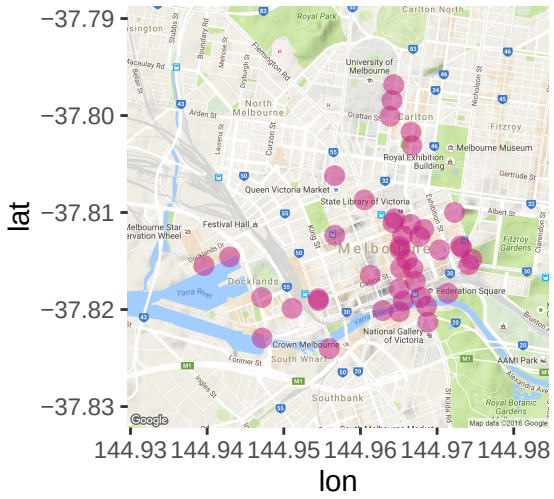
The `ggplot2` package has some basic maps included. To get the map of Australia we need to subset the world map based on the longitude and latitude of the station locations. Maps like this can be considered to be polygon data. We use these polygons as the background to the points corresponding to station locations.

```
# Read stations data
stations <- read_table("../data/ghcnd-stations.txt",
  col_names=c("ID", "lat", "lon", "elev", "state", "name",
              "v1", "v2", "v3"), skip=353, n_max=17081)

oz <- map_data("world", xlim=range(stations$lon),
               ylim=range(stations$lat))
ggplot(oz, aes(x=long, y=lat)) + geom_path(aes(group=group)) +
  coord_quickmap() +
  geom_point(data=stations, aes(x=lon, y=lat),
             colour="red", alpha=0.5)
```



We are next going to get the sensor locations, and plot these on a google map of Melbourne. You need to go to the Melbourne Open Data Portal site and export the locations as a csv file.

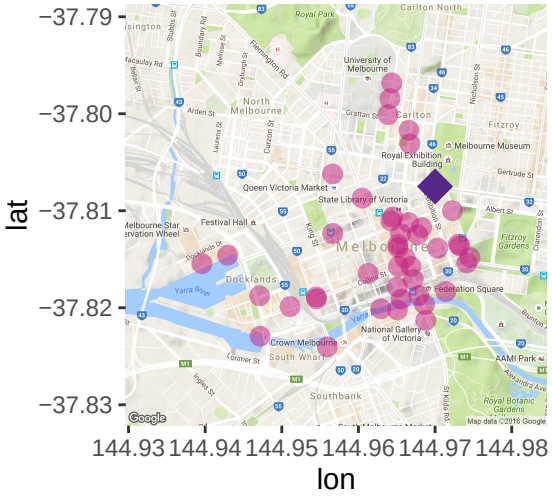
```
# Get pedestrian sensor locations
ped_loc <- read_csv("../data/Pedestrian_Sensor_Locations.csv")

melb <- get_map(location=c(mean(range(ped_loc$Longitude)),
                           mean(range(ped_loc$Latitude))), zoom=14)
ggmap(melb) + geom_point(data=ped_loc,
                         aes(x=Longitude, y=Latitude),
                         colour="#c51b7d", alpha=0.5, size=3)
```



Our next step is to extract the nearest weather station near downtown.

```
# Choose a weather station close to pedestrian sensors
melb_stns <- stations %>% filter(lon > min(ped_loc$Longitude),
                                 lon < max(ped_loc$Longitude),
                                 lat > min(ped_loc$Latitude),
```

```
                                  lat < max(ped_loc$Latitude))

ggmap(melb) + geom_point(data=ped_loc,
                         aes(x=Longitude, y=Latitude),
                         colour="#c51b7d", alpha=0.5, size=3) +
  geom_point(data=melb_stns, aes(x=lon, y=lat),
             colour="#542788", size=6, shape=18)
```



Combining pedestrian sensor data and weather data

Only a selection of sensors are used, ones that have complete records for 2013 and 2014. And only weather records for these years are used, partly because that weather station values for 2015 are mostly missing. We would need to pull weather data from the next nearest station to examine 2015 data.

We pulled the sensor data directly from the portal using this R code

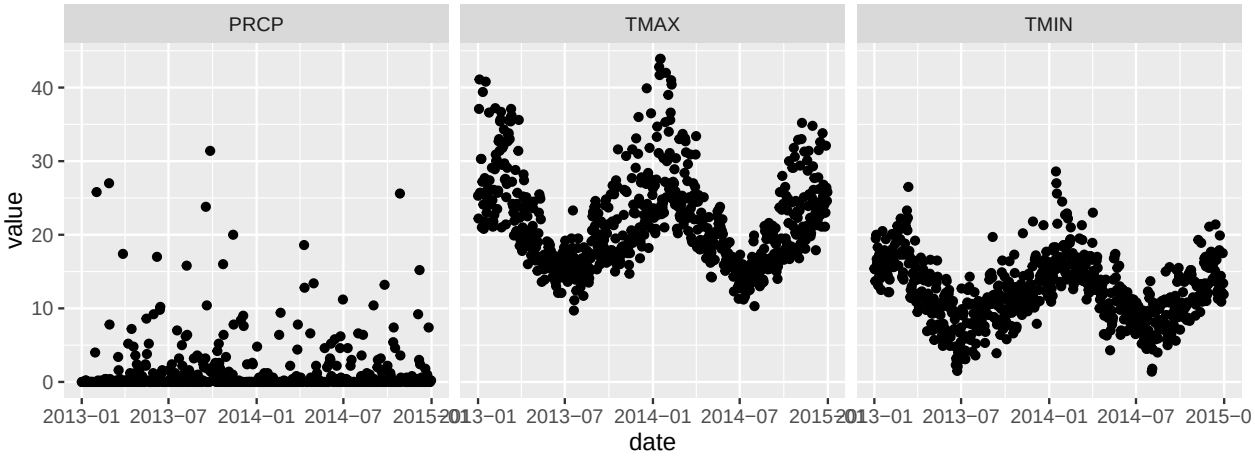
```
library(jsonlite)
limit <- 1453000 # all the up-to-date records need to be retrieved
web_add <- "https://data.melbourne.vic.gov.au/resource/mxb8-wn4w.json?"
ped_url <- paste0(web_add, "$limit=", limit)
pedestrian <- fromJSON(ped_url) # without api token
pedestrian <- tbl_df(pedestrian)
colnames(pedestrian) <- c("date_time", "day", "id", "mdate", "month", "count", "sensor_id", "sensor_nam
pedestrian <- pedestrian %>%
  mutate(date = as.Date(paste(pedestrian$mdate,
                              pedestrian$month,
                              pedestrian$year, sep="-"),
                        "%d-%b-%Y", tz = "AEST"),
         count = as.integer(count), sensor_id = factor(sensor_id))
```

and then wrote the data to file for use in this class.

The weather data was extracted from the yearly gzipped, csv files, using code like this:

```
t2013 <- read_csv("2013.csv.gz", col_names=FALSE)
t2013 <- t2013 %>% filter(X1 == melb_stns$ID)
```

Some additional pre-processing of the weather data is needed - month, day and year variables are extracted from the date column - a new date variable is created that is recognised as a date by R - temperature is converted into Celsius, precipitation is converted into mm - Missing precipitation values were substituted with 0.




```
#  [1] "Bourke Street Mall (South)"
#  [2] "Melbourne Central"
#  [3] "Town Hall (West)"
#  [4] "Princes Bridge"
#  [5] "Flinders Street Station Underpass"
#  [6] "Webb Bridge"
#  [7] "Southern Cross Station"
#  [8] "Victoria Point"
#  [9] "Waterfront City"
# [10] "Flagstaff Station"
# [11] "Sandridge Bridge"
```

Your group will be assigned one sensor to work with. Subset your data to have only the records for this one sensor, in order to do the questions for the lab. (My example code uses)

Question 1

a. Why was precipitation, max and min temp divided by 10?
b. Plot the minimum vs maximum temperature and describe the relationship. Why might it not be a good idea to use both of these in a model?
c. Replacing missing values with 0 is generally a REALLY BAD idea. Why is it reasonable for precipitation?
d. Three new variables were created, `high_prcp`, `high_tmp`, `low_tmp`. Explain why you think Dr Cook did this? And why these variables rather than the original will be used in the forthcoming model.
e. Make a plot of 3pm each day in January, and pedestrian `count` by `high_tmp`. You choose what you think is the best plot to make. What do you learn from the plot?
f. Make a time series plot of pedestrian counts, faceted by month and day. What do you learn about the pedestrian traffic at your sensor location?

Question 2

We are going to fit a Poisson model with

- Response: `count`
- Explanatory: `day`, `time`, `month`, `high_tmp`, `low_tmp`, `high_prcp`

Three-way interactions between `day`, `time`, `month` are included.

HEADS UP: The model will take a few minutes to fit, and to compute the predicted values. Start it running and then go get a coffee or tea.

a. Why use a Poisson model?
b. What does the three way interaction model allow for in the pattern of counts over day, time and month?
c. How many estimates need to be made in this model? Do you have enough data to do all of this estimation?
d. Examine the estimates, and significance for the three weather variables. How does high temperature affect the pedestrian traffic? Or does it have any effect?
e. Make a plot of the fitted values by time, faceted on month and day. Colour points by `high_temp`. What do you learn about the effect of high temperature on estimated pedestrian traffic?
f. Compute the difference between pedestrian count, for a Wednesday, 3pm, in January, for a `hot` vs `not` day. (Take precipitation to be `none` and `low_temp` to be `not`.)

Question 3

a. Plot the observed vs fitted values. How good is your model?
b. Plot the residuals against fitted values. What days, times and months have the largest residuals? (These would be times that the model doesn't predict well.)
c. Explore reasons for these misfits, using the internet, or what you know about Melbourne.

```
library(plotly)
p <- ggplot(ped_weath_bsm_aug, aes(x=.fitted_exp, y=.resid,
  label=paste(month, day, mday, year.x, "-", time))) +
  geom_point()
ggplotly(p)
```

TURN IN

- Your `.Rmd` file
- Your Word (or pdf) file that results from knitting the Rmd.
- Make sure your group members are listed as authors, one person per group will turn in the report
- DUE: Wednesday after the lab, by 7am, loaded into moodle

Resources

- Pedestrian count data
- Menne, M.J., I. Durre, B. Korzeniewski, S. McNeal, K. Thomas, X. Yin, S. Anthony, R. Ray, R.S. Vose, B.E.Gleason, and T.G. Houston, 2012: Global Historical Climatology Network - Daily (GHCN-Daily), Version 3. [indicate subset used following decimal, e.g. Version 3.12]. NOAA National Climatic Data Center. http://doi.org/10.7289/V5D21VHZ [9/9/2016].