Stats 598z: Homework 3

Due before class on Thursday, Feb 16

Important:

`R` code, tables and figures should be part of a single .pdf or .html files from `R Markdown` and `knitr`. See the class reading lists for a short tutorial.
Include `R` commands for all output unless explicitly told not to.

If you collaborated with anyone else, mention their names and the nature of the collaboration

1 Problem 1: Dataframes and `ggplot2` [100pts]

(a) Install the package `maps` and load it into your session of `R` [3pts]

(b) The function `map_data()` returns a dataframe containing information for a map, the choice of map determined by the `map` argument. Call this function, setting `map` to `"state"`. What is the dimensionality of the returned dataframe? What are the names of its columns? [4pts]

(c) The `unique()` function returns the unique elements of an vector. Print the number of unique values of the column `region` of the dataframe returned in the previous step. [3pts]

(d) The columns `lat` and `long` of the dataframe correspond to latitude and longitude. Use `ggplot` to plot the dataframe with the x-axis as the longitude and the y-axis as lattitude. States should have different color. Each (latitude, longitude)-pair should be a 'point'. [10pts]

(e) Now connect the points from the previous slide with lines. Is `geom_line` or `geom_path` appropriate? Explain the difference and show the plot. [10pts]

(f) Choose any two states. Plot the (latitude,longitude)-pairs only for those two states, connecting them with lines as before. [5pts]

(g) Repeat the last step, first using the `sample()` function to randomly permute the rows.

(h) A much nicer approach to plotting is to use the `geom_map` geometry. From here onwards, we will call the result from part (b) as `states_map`. Then this command will look like:

```
ggplot() + geom_map(map = states_map, map_id=states_map$region, data = states_map, aes(fill=group))
+ expand_limits(x = states_map$long, y = states_map$lat)
```

Make sure this works for your choice of variable names, and show the output. [10pts]

(i) You can look at the documentation of `geom_map`, but to keep things simple we will focus only on the `aes` part. Above we have chosen to fill each state according to the `group` column. Note that `group` takes 63 values which is more than the number of states. To avoid any possible mistakes, add a new column to `states_map`, assigning each row a unique integer according to its associated state. You can use a for loop. Plot the result. [10pts]

(j) The `state` dataset contains information for US states, you can load it by `data(state)`. `state.name` gives the name of states, and `state.x77` gives some statistics for the states. The fifth column gives the number of murder arrests per 100,000 people for different states. Repeat the last step, now assigning each row its associated murder arrest count (call the column `"InfoValue"`. Since one dataframe capitalizes the state names, and the other doesn't, you might want to use to `tolower()` function to convert strings to lower case. Again, you can use for loops if you want to. WARNING: the order of states in the two datasets is not necessarily the same. [10pts]

(k) In this and the next question, we will plot two maps side by side using `facet_grid`. You can choose any two columns of `state.x77`, I recommend picking two that have similar ranges (or normalize them to be so). From the example in the class slides, `facet_grid` expects a column which determines which panel the corresponding row belongs to. To do this, add one more column to `states_map`: call it `"InfoType"` and set it to `"Murder"` for each row. Then make a copy of `states_map`, setting `"InfoType"` to (e.g.) `"Grad"` and filling `"InfoValue"` with the corresponding column of `state.x77`. Stack these two dataframes on top of each other using `rbind()`. (Later, we will see easier ways of doing this using the `reshape` package.) [10pts]

(l) Plot this using the `facet_grid` layer, with the facet determined by `"InfoType"`. [5pts]

(m) Go back to part (j) where we have only one facet. Pick another column of `state.x77`: we will represent it as a circle located at the center of each state with radius proportional to value. Accordingly, create a new dataframe, with columns `"State", "MeanLat", "MeanLong"` and `"StatValue"`. Here `("MeanLat","MeanLong")` are the mean (latitudes,longitudes) of the rows in `states_map` associated with each state. You can use for loop to calculate these. [10pts]

(n) We used for loops in the previous parts. More compact is to use the `sapply` function. This expects two inputs: a vector `X` and a function `fun`, and returns a vector of the same length as `X`, obtained by applying `fun` to each element of `X`. We will set `X` to be the set of unique state names in `states_map`. The function `fun` expects a single input (a state name), and calculates and returns the mean latitude and longitude of that state. Write down the function `fun`, and print the output of applying to `states_map`. This should be the same as the previous subquestion. [10pts]