Stat 610 Homework 3

Due Friday, October 2, 11:59pm. You may work in groups of up to 3.

Background

You have a collaborator who studies American politics. She is interested in the relationship between how active a member of congress is and their voting patterns. In particular, she is interested in describing how often a given member's vote on a bill aligns with the vote of the chamber overall: If the measure passed, did the congressperson vote for it? If it failed, did the congressperson vote against it?

She has information on voting records and some information about the votes that were taken by the senate in 1999. At the moment she doesn't have a good way of defining how active a congressperson is, but as a first pass, she would like to measure the activity level by how frequently his or her name comes up in descriptions of bills.

She has three data sets available:

- `bills.csv`: Has information on everything the senate voted on in 1999. Includes the fraction of the senate needed for the measure to pass, text describing the measure, and an id.
- `members.csv`: Has information on the senators who served that year. This includes their names, party, and a unique id.
- `votes.csv`: Has the information about how each senator voted on each measure. Rows are senators, labeled by the id in `members.csv`, and columns are bills, labeled by the id in `bills.csv`.

All of the information required to do the analysis is present in these datasets, but it is not in a convenient form, and your collaborator needs help computing the variables she wants and getting the data into shape so that it can be analyzed.

Assignment

Your assignment is as follows:

1. Download the three datasets, `bills.csv`, `members.csv`, and `votes.csv`, and read them in to R using the `read.csv` function. For `votes.csv`, I highly recommend using the optional argument `row.names = 1`, which will give you a data frame containing just vote types, and ids stored as row names. I also recommend using `stringsAsFactors = FALSE`.
2. Create the measure of activity level described above. That is, for each senator, find the number of times his or her last name appears in the `subject` column of `bills.csv`. Add this as a new column to the the data frame containing the information in `members.csv`.
3. For each bill, find the number of "Yea" votes. To do this, you should create a small function that takes as its argument a vote vector (e.g., one of the columns of the `votes` data frame)

and returns the number of "Yea" votes. Apply this function to the `votes` data to get a data frame with one column giving the number of Yea votes and another column giving the id of the bill. Use one of the `**ply` functions from `plyr` or a `dplyr` function that we talked about in class (which one should it be?).

4. Before you can decide whether the bills passed or not, you need to find how many votes were required for passage. This is not entirely straightforward because different bills have different requirements: some require 50 votes, some require 67 votes, and some require 60 votes. The number of votes required for each bill is found in the `requires` column of `bills`, but it is formatted as a string. Add a new column to the `bills` data frame giving the number of Yea votes required for passage. As in the last part, one way to do this is to make a small function that takes the string version of the fraction of votes required, returns the number of Yea votes required, and apply that function to the `requires` column of `bills`.

   N.B.: It's possible that someone else has written a function that takes strings like `'1/2'` and turns them into numbers. If you know about one, feel free to use that instead.

5. Finally, compute the fraction of the time each senator's vote was aligned with the overall outcome of the bill (that is, the senator voted Yea and the bill passed or the senator didn't vote Yea and the bill didn't pass), and add it as a new column to the `members` data frame.

   There are several ways to do this, but one strategy is to create a function that takes one data frame describing a senator's votes (one column for how the senator voted on the bill, one column for bill id), another data frame describing whether the bill passed (one column for whether the bill passed, one column for the bill id). The function can then merge those data frames on bill id and use the merged data frame to find how often the senator's vote was aligned with the overall passage of the bill. You can then apply that function to the `votes` data frame (with one of the `**ply` functions or with `dplyr`, which one?) to get the alignment measure.

   N.B.: The merge seems like it might be overkill, but there's nothing inherent in the way the data is set up that ensures that the order of the bills in `votes` and `bills` is the same. Because of that, it's safer to do the merge explicitly instead of assuming that the bills take the same order.

6. At this point, your `members` data frame should have one column with the activity measure for that senator and another column describing how often that senator's vote aligned with the passage or failure of the bill. Now we can answer the initial question!

   Within each party (given to you in the `party` column of `members`), fit a linear model with activity level as the predictor and vote alignment as the response. You should use one of the `**ply` or `dplyr` functions here again: think about how the data needs to be split, what function you need to apply, and what format the output should be in to decide which of the `**ply` functions you should use and what their arguments should be.

   Print out the output from each of the linear models and describe the results.

Submission parameters

– Submit an `.Rmd` file with your code and a description of what it is doing.