DS5110 HW 3 - Due Apr. 3

Kylie Ariel Bemis

3/20/2017

Instructions

Create a directory with the following structure:

- `hw3-your-name/hw3-your-name.Rmd`
- `hw3-your-name/hw3-your-name.pdf`
- `hw3-your-name/package-name.tar.gz`

where `hw3-your-name.Rmd` is an R Markdown file that compiles to create `hw3-your-name.pdf`, and `package-name.tar.gz` is your solution for Problem 10.

Do not include data in the directory. Compress the directory as `.zip`.

Your solution should include all of the code necessary to answer the problems. All of your code should run (assuming the data is available). All plots should be generated using `ggplot2`. Missing values and overplotting should be handled appropriately. Axes should be labeled clearly and accurately.

To submit your solution, create a new private post of type "Note" on Piazza, select "Individual Student(s) / Instructor(s)" and type "Instructors", select the folder "hw3", go to Insert->Insert file in the Rich Text Editor, upload your `.zip` homework solution. Title your note "[hw3 solutions] - your name" and post the private note to Piazza. **Be sure to post it only to instructors**

Part A

Problems 1–3 use the `BostonHousing` data from the `mlbench` package. Install the `mlbench` package from CRAN and use `data(BostonHousing)` to load the dataset.

Problem 1

Fit a model that predicts per capita crime rate by town (`crim`) using only *one* predictor variable. Use plots to justify your choice of predictor variable and the appropriateness of any transformations you use. Print the values of the fitted model parameters.

Problem 2

Plot the residuals of the fitted model from Problem 1 against the predictor variable already in the model *and* against other potential predictor variables in the dataset. Comment on what you observe in each residual plot. (You do not need to include plots for predictor variables not in the model where you observe no systematic patterns in them.)

Problem 3

Fit a new model for predicting per capita crime rate by town, adding or removing variables based on the residual plots from Problem 2.

Part B

Problem 4

Write a function that performs cross-validation for a linear model (fit using `lm`) and returns the average root-mean-square-error across all folds. The function should take as arguments (1) a formula used to fit the model, (2) a dataset, and (3) the number of folds to use for cross-validation. The function should partition the dataset, fit a model on each training partition, make predictions on each test partition, and return the average root-mean-square-error.

Problem 5

Use your function from Problem 4 to compare the models you used from Part A with 5-fold cross-validation. Report the cross-validated root-mean-square-error for the models from Problems 1 and 3. Which model was more predictive?

Part C

Problems 6–8 use the text of 56 major speeches by Donald Trump from June 2015 through November 2016. Download the data from https://github.com/PedramNavid/trump_speeches. (Use either the "full_speech.txt" file or the "speech_xx.txt" files but not both, as the former contains all of the text of the latter. If you use the "speech_xx.txt" files, you should skip the first line of each file.) Use the `read_lines()` function from the `readr` package to import the data into R.

Problem 6

Import the text from all 56 Donald Trump speeches into R and tokenize the data into a tidy text data frame, using *words* as tokens. After removing stop words and the word "applause", plot the top 15 most common words used in Trump's speeches.

Problem 7

Re-tokenize the text of all 56 Donald Trump Speeches into a new tidy text data frame, using *bigrams* as tokens. Remove each bigram where either word is a stop word or the word "applause". Then plot the top 15 most common bigrams in Trump's speeches.

Problem 8

We would like to do a sentiment analysis of Donald Trump's speeches. In order to make sure sentiments are assigned to appropriate contexts, first tokenize the speeches into bigrams, and then filter out all bigrams where the first word is any of "not", "no", or "never".

Now consider only the second word of each bigram. After filtering out the words "applause" and "trump", create a plot of the 10 most common words in Trump's speeches that are associated with each of the 10 sentiments in the "nrc" lexicon.

Part D

Problem 9

Create an S3 class called `tidy_corpus` which inherits from a tibble (`tbl_df`). A `tidy_corpus` is a tidy text data frame that always has at least one column called `token`, which gives the tokens, and an attribute called `token_type` which gives the type of token. A `tidy_corpus` object may also have an additional attribute called `n` when `token_type` is "ngrams".

Create a constructor function that creates `tidy_corpus` objects called `tidy_corpus(src, token, ...)` which takes three parameters:

- `src` is either a character vector or a directory of text files
- `token` gives the kind of tokenization to be performed
- `...` are additional arguments passed to `unnest_tokens()`

Write a `plot()` method for the `tidy_corpus` class that plots the top `n` most common tokens in the corpus where `n` is a user parameter defaulting to 10. Include an optional parameter for removing stop words. The removal of stop words should support at least words and bigrams.

Test your class and methods on the data in Part C.

Problem 10

Create an R package for the class and methods you created in Problem 9. For full credit, it should pass `R CMD check` without errors. (Warnings are okay. You do not need to include documentation.) Build the package and include the `.tar.gz` compressed file in your homework directory.