STA314: Assignment 1, Fall 2020

This assignment is **Due 27 November at 12pm**.

Please submit your answers via Quercus as a *single* `.zip` file. This zip file should include:

- One (1) .R file called `functions.R` that contains all of the R functions you wrote.
- One (1) .R file called `make_tidy.R` that contains a script the defines and registers the `parsnip` engine for the functions defined in `functions.R`.
- A pdf of the *compiled* vignette.

The marking scheme is as follows:

- 40% for the functions in `functions.R` and the tidymodels wrapper defined in `make_tidy.R` passing some unit tests. (See below for details)
- 20% for complete documentation (as described in the Week 4 tutorial) for all functions. This should include a simple use case for each function.
- 40% for the vignette. It should clearly step through the process of using your functions to complete the task using non-technical language where appropriate. All plots should have clear captions, axis names, and legends.

Please note: The raw markdown file **is not required**.

Please note: Please read the instructions *very carefully*. There is a restriction on the packages you are allowed to use in this assignment.

The task

Your task for this assignment is to program up functions to compute the logistic Lasso using coordinate descent on the the penalized iteratively reweighted least squares algorithm. *Please note:* You will need to modify the derivation of coordiante descent to allow for weights!

You will then build the helper functions required to add your algorihm as a `parsnip` model so that it can be used in a `tidymodels` workflow.

You are to then to write a short vignette that explains how to use your function within a tidymodels workflow to perform classification. This should include how to tune the model.

This vignette should **not** call any of the functions in the file `functions.R` directly, but should instead have the following code (or its equivalent) somewhere near the beginning

```
source("functions.R")
source("make_tidy.R")
```

From that point forward, the vignette should use the functions you have written **only** through the tidymodels interface you have built.

The functions

You only need to write two main function (although you're welcome to use helper functions as long as they are documented appropriately). It should have the following signature:

`ret <- fit_logistic_lasso(x, y, lambda, beta0 = NULL, eps = 0.0001, iter_max = 100)` where:

- `x`: matrix of predictors (not including the intercept)

- `y`: vector of data
- `lambda`: penalty
- `beta0`: initial guess
- `eps`: parameter for stopping critereon
- `iter_max`: maximium number of iterations
- `ret` A list containing the members `intercept`, `beta`, and `lambda`

```
ret <- predict_logistic_lasso(object, new_x)
```

where:

- `object`: Output from `fit_logistic_lasso`
- `new_x`: Data to predict at (may be more than one point!)
- `ret` A list containing the intercept and beta

Unit tests

The exact unit tests that will be used for marking will not be released.

The purpose of these tests is to ensure that the functions work as indicated. Below is some skeleton code for a unit test for `fit_logistic_lasso`

```
# simulte some data


lambda <- 0.3

# compute coefficients and intercept using glmnet and store as int_true and
# beta_true.

ret <- fit_logistic_lasso(x=x,y=y,lambda = 0.3)

if (abs(int_true - ret$intercept) > 0.01) { ## this may not be the tolerance!
 print(glue::glue("Test failed. Expected intercept {int_true} but got
                  {ret$intercept}\n")
}

if (mean(abs(beta_true - ret$beta)) > 0.01) { ## this may not be the tolerance!
 print(glue::glue("Test failed. Expected computed beta had an error of
                  {mean(abs(beta_true - ret$beta))}.\n")
}
```

The unit tests will test both functions **and** will test the tidymodels interface.

I **strongly** recommend that you write your own version of these tests to check your code. This is a situation where you can (and should!) collaborate with fellow students and compare your test code. As always, *you are not allowed to collaborate on any other aspect of this assignment.*

Which packages can you use

You can use any packages included when you run the following commands:

```
library(tidyverse)
library(tidymodels)
```

Note these are *metapackages* that include a whole host of other packages including `dplyr`, `ggplot`, and `parsnip` among others.

In particular, you **may not** use any implementations of the logistic lasso in your submitted code (eg you can't use the one in `glmnet`.). You may, however, find these implementations useful when you write your own tests. But please don't submit this work.