A2

TONGFEI ZHOU

2020/11/25

In this assignment, I mainly realize the algorithm of Logistic regression with LASSO penalized term and using coordinate descent method in updating the beta parameters.

First we will source the make_tidy file so that the me-written algorithms fit_logistic_lasso and predict_logistic_lasso will be added to parsnip package.

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

```
## Information for `logistic_lasso`
##  modes: unknown, classification
##
##  engines:
##    classification: fit_logistic_lasso
##
##  arguments:
##    fit_logistic_lasso:
##       penalty --> lambda
##
##  no registered fit modules.
##
##  no registered prediction modules.
```

The model information is printed above, however it is the intermediate results and so the rest finalized part is cotinued to be done in the make_tidy.R file.

Now we get our model, and I simulate the data as in tutorial 9 solution as follows:

```
set.seed(8448)
n = 1000
dat <- tibble(x = seq(-3,3, length.out = n),
              w = 3*cos(3*seq(-pi,pi, length.out = n)),
              y = rbinom(n,size = 1, prob = 1/(1 + exp(-w+2*x)) )%>% as.numeric %>% factor,
              cat = sample(c("a","b","c"), n, replace = TRUE)
              )

split <- initial_split(dat, strata = c("cat"))

train <- training(split)
test <- testing(split)

rec <- recipe(y ~ . , data = train) %>%
  step_dummy(all_nominal(), -y) %>% step_zv(all_outcomes()) %>%
```

```
  step_normalize(all_numeric(), -y) %>%
  step_intercept()

head(dat)
```

```
## # A tibble: 6 x 4
##       x     w y     cat
##   <dbl> <dbl> <fct> <chr>
## 1 -3    -3    0     c
## 2 -2.99 -3.00 1     c
## 3 -2.99 -3.00 1     a
## 4 -2.98 -3.00 1     a
## 5 -2.98 -2.99 1     a
## 6 -2.97 -2.99 1     c
```

As we can see from the piece of dat, it is a model with response Y as categorical variable with only possible values of 0 and 1, and three predictors, x, w as numeric predictors and cat as a categorical variable with values 'a', 'b', 'c'.

Then I randomly choose λ to be 0.1 and check the quality of the prediction result.

```
lambda = 0.1
spec <- logistic_lasso(penalty=lambda) %>% set_engine("fit_logistic_lasso")

logistic_lasso_result <- workflow() %>% add_recipe(rec) %>% add_model(spec) %>% fit(train)


predict(logistic_lasso_result, new_data = test) %>% bind_cols(test %>% select(y)) %>%
  conf_mat(truth = y, estimate = .pred_class)
```

```
##            Truth
## Prediction   0   1
##          0 111  13
##          1  11 114
```

As we can see above, the prediction results is not bad at all since with both cases there are only 13 and 11 errors.

Now we check that we got the answers correct by comapring with glm like we did in tut9 sol. Also this is the unit test we will run.

```
# Make the data
ddat<- rec %>% prep(train) %>% juice

ff = logistic_reg(penalty = lambda, mixture = 1) %>%
  set_mode("classification") %>%
  set_engine("glm") %>%
  fit(y ~ ., family = "binomial", data = ddat)


# We also print the the compare error table so that we can make conclusion in the bottom easier
compare = ff %>% tidy %>% select(term, estimate) %>%
  mutate(logistic_lasso_estimate = c(logistic_lasso_result$fit$fit$fit$intercept,logistic_lasso_result$
```

```
compare <- compare[-c(2),]
compare
```

```
## # A tibble: 5 x 4
##   term        estimate logistic_lasso_estimate      err
##   <chr>          <dbl>                  <dbl>    <dbl>
## 1 (Intercept)   -0.226                 -0.224 -0.00162
## 2 x             -3.20                  -3.19  -0.00962
## 3 w              1.85                   1.84   0.00637
## 4 cat_b         -0.171                 -0.168 -0.00339
## 5 cat_c         -0.192                 -0.188 -0.00331
```

```
int_true <- compare$estimate[1] # The intercept value computed by functions in R
beta_true <- compare$estimate[-1] # The beta value computed by functions in R

ret_intercept <- compare$logistic_lasso_estimate[1]
ret_beta <- compare$logistic_lasso_estimate[-1]

if (abs(int_true - ret_intercept) > 0.01) {
  print(glue::glue("Test failed. Expected intercept {int_true} but got
                   {compare$logistic_lasso_estimate[1]}\n"))
}

if (mean(abs(beta_true - ret_beta)) > 0.01) {
  print(glue::glue("Test failed. Expected computed beta had an error of
                   {mean(abs(beta_true - ret_beta))}.\n"))
}
```

As we can see the error is quite small, so Nice~. (And we past the unit test with $\lambda = 0.1$ in this case.)

Also we need to make sure that our predict works too!

```
test_dat <- rec %>% prep(train) %>% bake(test)
glm_pred <- (predict(ff, test_dat) == 1) %>% as.numeric
preds <- predict(logistic_lasso_result, new_data=test) %>% bind_cols(glm_pred = glm_pred)
if(any(preds$.pred_class != preds$glm_pred)){
  warning("There are some predictions that are not the same with glmnet.")
}else{
  print("All the predictions are the same with glmnet.")
}
```

```
## [1] "All the predictions are the same with glmnet."
```

Since all the predictions are the same with glmnet, our prediction works too and so the algorithm written by me is correct in this unit test!