STAT 471: Midterm Review Practice Problems (Solution)

Shuxiao

Oct. 21, 2021

Contents

Note: this document is adapted from last year's midterm.

Instructions

- This exam is open-book / open-notes / open-internet. However, it is individual work. Communication among students is prohibited.
- Please complete your homework in R Markdown, using this document as a starting point. Show your R code using code chunks and add your text answers using **bold text**.
- When you are ready to submit, please compile your R Markdown file into a PDF. Then, submit this PDF through Canvas.
- While base R programming is acceptable, I strongly encourage you to use the `tidyverse` to complete your assignment. However, points will not be deducted if you use base R programming.
- In addition, please make sure to have clear labels and sensible titles on your plots.
- **Make sure that `readmission_clean.csv` and `plot_glmnet.R` are in your working directory before beginning the exam.**

Introduction: Predicting readmission for diabetes patients

Background and goals

Diabetes is a chronic medical condition affecting millions of Americans, but if managed well, patients can lead relatively normal lives. However, if improperly managed, diabetes can lead to patients being continuously admitted and readmitted to hospitals. Hospital readmissions represent a failure of the health system to provide adequate support to the patient and are extremely costly to the system. The goals are therefore to

1. identify important factors associated with readmission, and
2. predict whether a given patient will be readmitted.

Readmission data

In this exam, we will investigate a dataset originally from the Center for Clinical and Translational Research at Virginia Commonwealth University, covering diabetes patients across 130 U.S. hospitals from 1999 to 2008. Three former STAT 471 students Spencer Luster, Matthew Lesser and Mridul Ganesh brought this data set into the class through their final project. In this exam, we will use a cleaned subset of this data.

First, let's load a few libraries:

```
library(kableExtra)                     # for printing tables
library(cowplot)                        # for side by side plots
library(glmnetUtils)                    # to run ridge and lasso
library(lubridate)                      # for dealing with dates
library(maps)                           # for creating maps
source("../../functions/plot_glmnet.R") # for lasso/ridge trace plots
library(tidyverse)                      # for everything else
```

Let's load the data:

```
readmission = read_csv("../../data/readmission_clean.csv")
readmission
```

```
## # A tibble: 99,492 x 26
##    race            gender age_group num_outpatient num_inpatient num_emergency
##    <chr>           <chr>  <chr>              <dbl>         <dbl>         <dbl>
##  1 Caucasian       Female 80+                    0             0             0
##  2 Caucasian       Female 80+                    0             0             0
##  3 Caucasian       Male   20-59                  0             0             0
##  4 AfricanAmerican Female 20-59                  0             0             0
##  5 Caucasian       Male   20-59                  0             0             0
##  6 AfricanAmerican Male   60-79                  0             0             0
##  7 Caucasian       Female 20-59                  0             0             1
##  8 Caucasian       Male   80+                    0             0             0
##  9 Caucasian       Male   60-79                  0             0             0
## 10 AfricanAmerican Female 60-79                  0             0             0
## # ... with 99,482 more rows, and 20 more variables: num_medications <dbl>,
## #   num_diagnoses <dbl>, adm_source <chr>, adm_type <chr>,
## #   time_in_hospital <dbl>, num_lab_procedures <dbl>, num_procedures <dbl>,
## #   discharge <chr>, max_glu_serum <chr>, A1Cresult <chr>, med_changed <chr>,
## #   med_prescribed <chr>, insulin <chr>, metformin <chr>, glimepiride <chr>,
## #   glipizide <chr>, glyburide <chr>, pioglitazone <chr>, rosiglitazone <chr>,
## #   readmitted <dbl>
```

Each row corresponds to a hospital admission of a patient. There are 26 total variables, described below:

Demographic variables

- `race`: patient's race
- `gender`: patient's gender
- `age_group`: patient's age group

Medical history

- `num_outpatient`: number of outpatient visits by the patient in the year prior to the current admission
- `num_inpatient`: number of inpatient visits by the patient in the year prior to the current admission
- `num_emergency`: number of emergency visits by the patient in the year prior to the current admission
- `num_medications`: number of total medications the patient has taken
- `num_diagnoses`: number of total diagnoses the patient has

Hospital admission details

- `adm_source`: who referred the patient to the hospital
- `adm_type`: type of admission
- `time_in_hospital`: length of stay in the hospital (in days)
- `num_lab_procedures`: number of lab procedures performed
- `num_procedures`: number of non-lab procedures performed
- `discharge`: where the patient was discharged

Clinical results

- `max_glu_serum`: results of glucose serum test
- `A1Cresult`: results of A1c test

Medication details

- `med_changed`: whether any medication was changed
- `med_prescribed`: whether any medication was prescribed
- `insulin`: type of change (if any) to insulin medication
- `metformin`: type of change (if any) to insulin medication
- `glimepiride`: type of change (if any) to glimepiride medication
- `glipizide`: type of change (if any) to glipizide medication
- `glyburide`: type of change (if any) to glyburide medication
- `pioglitazone`: type of change (if any) to pioglitazone medication
- `rosiglitazone`: type of change (if any) to rosiglitazone medication

Readmission indicator

- `readmitted`: whether the patient was readmitted to the hospital within 30 days of discharge

Train/test split

Let's subsample 10000 observations for training and subsample a non-overlapping 2000 observations for testing:

Table 1: Frequencies of admission source for patiens in the age of 0-19

adm_source	frequency
Emergency Room	55
Other	5
Physician Referral	23
Transfer from Home Health	1

```
set.seed(1)
n_total = nrow(readmission)
n_train = 10000
n_test = 2000
train_samples = sample(1:n_total, n_train)
test_samples = sample((1:n_total)[-train_samples], n_test)
readmission_train = readmission %>% filter(row_number() %in% train_samples)
readmission_test = readmission %>% filter(row_number() %in% test_samples)
```

1 Data wrangling and exploratory data analysis

First, let's do some exploratory data analysis on our training data `readmission_train`.

1. For people in the age of 0–19, what are the frequencies of `adm_source`? Present the result in a nice table.

```
readmission_train %>% filter(age_group=='0-19') %>% count(adm_source) %>% rename(frequency=n) %>%
  kable(format = "latex", row.names = NA,
        booktabs = TRUE, digits = 2,
        # col.names = c("County", "State", "Case-fatality rate"),
        caption = "Frequencies of admission source for patiens in the age of 0-19") %>%
  kable_styling(position = "center")
```

2. For each unique (gender, age_group) combination, calculate the sum of `num_outpatient`, `num_inpatient`, `num_emergency`, then calculate the total sum of those three variables. Tabulate the result in a fancy table.

```
readmission_train %>%
  group_by(gender, age_group) %>%
  summarise(total_num_outpatient=sum(num_outpatient),
            total_num_inpatient=sum(num_inpatient),
            total_num_emergency=sum(num_emergency)) %>%
  ungroup() %>%
  mutate(grand_total=total_num_outpatient+total_num_inpatient+total_num_emergency) %>%
  kable(format = "latex", row.names = NA,
        booktabs = TRUE, digits = 2,
        col.names = c("Gender", "Age", "Num outpatient", "Num inpatient",
                      "Num emergency", "Grand total"),
        caption = "Case numbers for each gender-age combination") %>%
  kable_styling(position = "center")
```

Table 2: Case numbers for each gender-age combination

Gender	Age	Num outpatient	Num inpatient	Num emergency	Grand total
Female	0-19	6	22	9	37
Female	20-59	604	1325	670	2599
Female	60-79	1059	1585	451	3095
Female	80+	452	836	210	1498
Male	0-19	0	30	1	31
Male	20-59	417	976	395	1788
Male	60-79	949	1370	293	2612
Male	80+	331	399	95	825

```
## ‘summarise()‘ has grouped output by ’gender’. You can override using the ‘.groups‘ argument.
```

3. What fraction of the patients in the training data were readmitted?

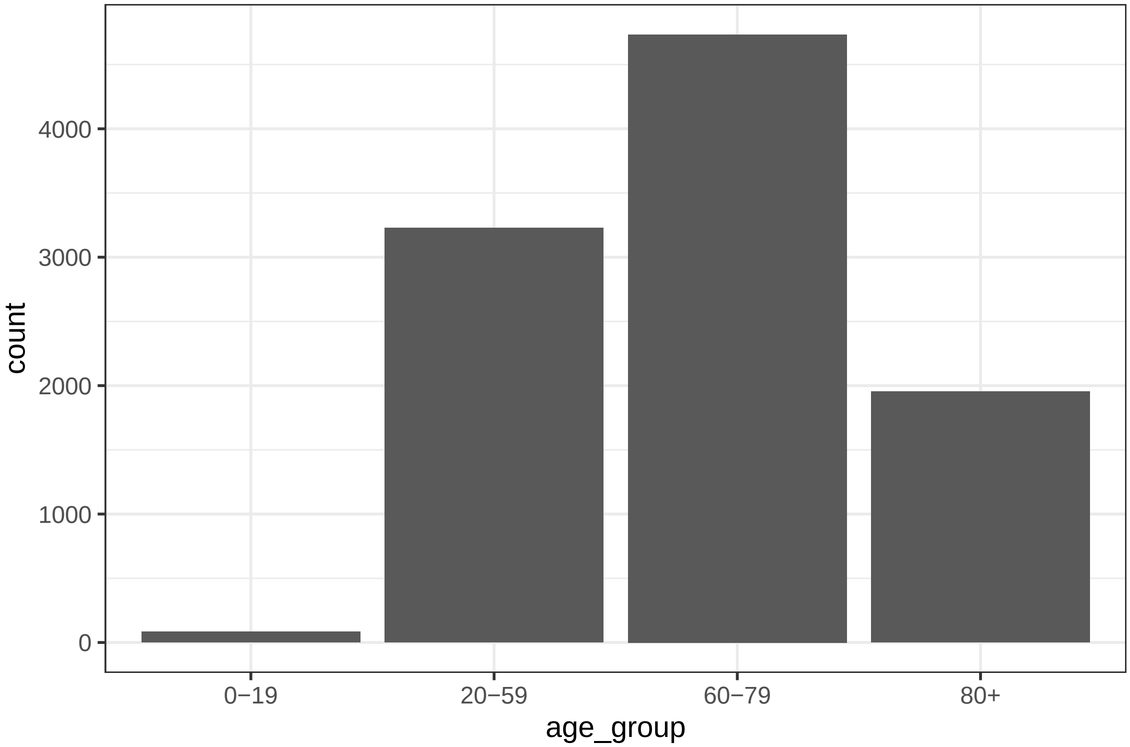
```
readmission_train %>% summarise(frac_readmitted = mean(readmitted))
```

```
## # A tibble: 1 x 1
##   frac_readmitted
##             <dbl>
## 1           0.117
```

We see that 0.117 of the patients in the training data were readmitted.

4. Produce a bar plot to display the breakdown of the patients by age group. What is the most prevalent age group in the training data?

```
readmission_train %>%
  ggplot(aes(x = age_group)) + geom_bar() + theme_bw()
```

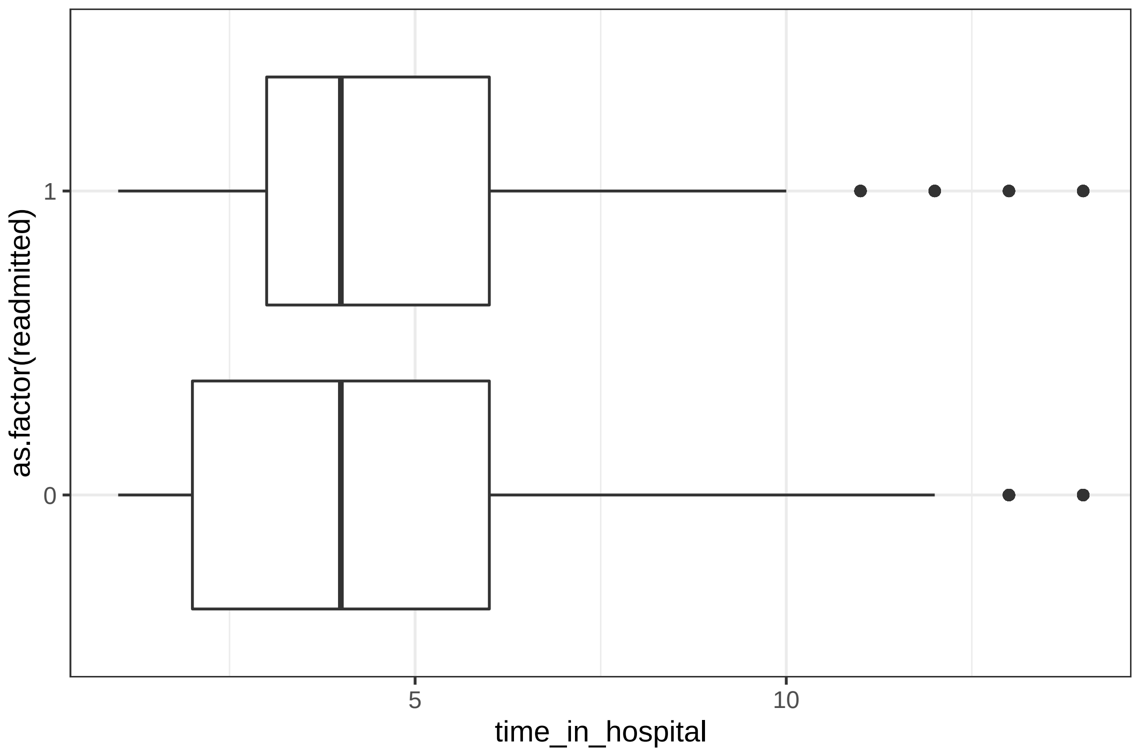


The most prevalent age group is 60-79.

5. Produce a plot to show the relationship between `time_in_hospital` and `readmitted`. Using `summarise`, compute the median time in hospital separately for patients that were not readmitted and for those that were. Do these suggest that readmission rates vary based on time in hospital, and if so, what is the direction of the relationship?

(Hint: It may be useful to convert `readmitted` to a factor using `as.factor(readmitted)`.)

```
readmission_train %>% ggplot(aes(x = time_in_hospital, y = as.factor(readmitted))) +
  geom_boxplot() + theme_bw()
```



```
readmission_train %>%
  group_by(readmitted) %>%
  summarise(median_time_in_hospital = median(time_in_hospital))
```

```
## # A tibble: 2 x 2
##   readmitted median_time_in_hospital
##        <dbl>                   <dbl>
## 1          0                       4
## 2          1                       4
```

The median number of days in the hospital was 4 days for both groups of patients. This fact, along with the plot above, suggests that time in hospital is not associated with readmission.

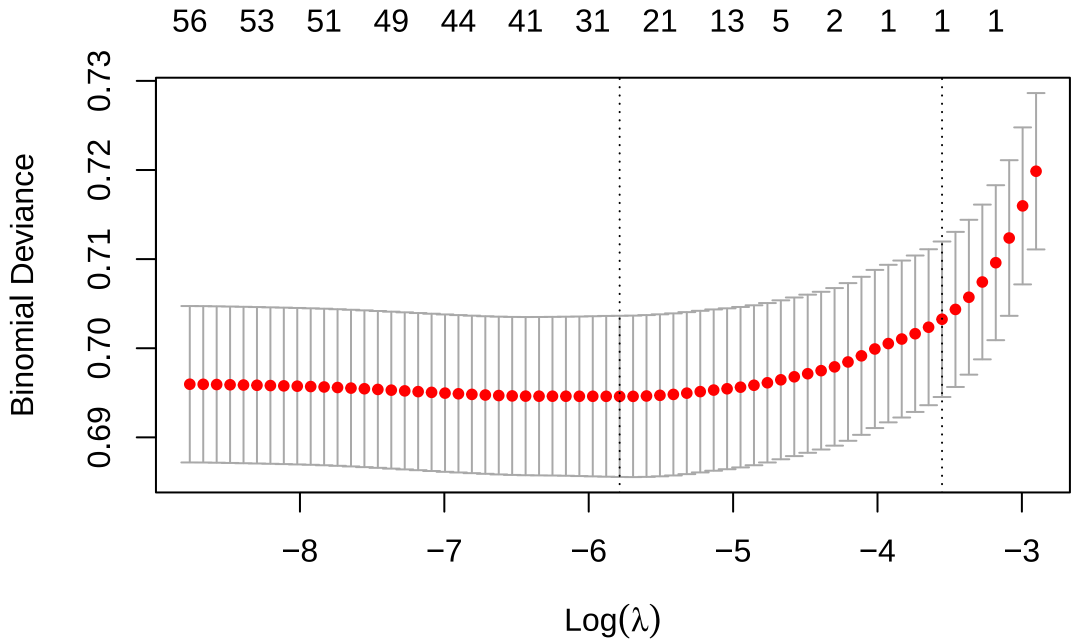
2 Prediction via regularized logistic regression

i. Fit a 10-fold cross-validated lasso logistic regression to the training data.

```
# run cross-validated logistic lasso regression
set.seed(3)
lasso_fit = cv.glmnet(readmitted ~ ., family='binomial',
                      alpha = 1,
                      nfolds = 10,
                      data = readmission_train)
```

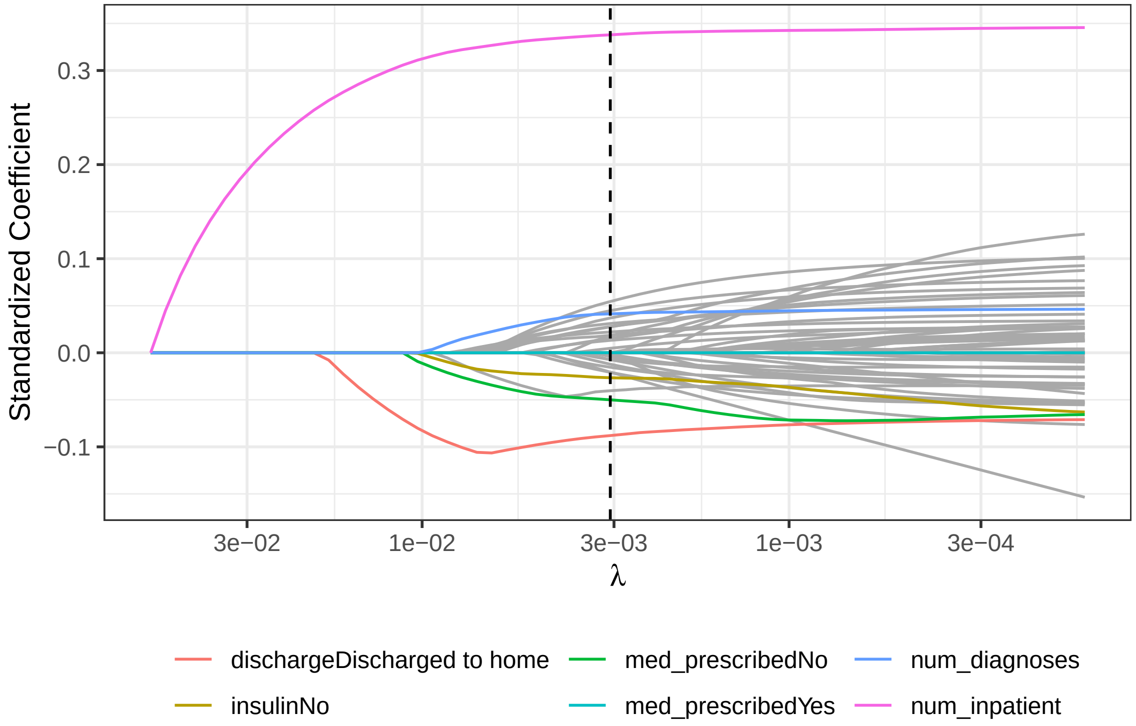
ii. Produce the CV plot, and give interpretations for this plot.

```
plot(lasso_fit)
```



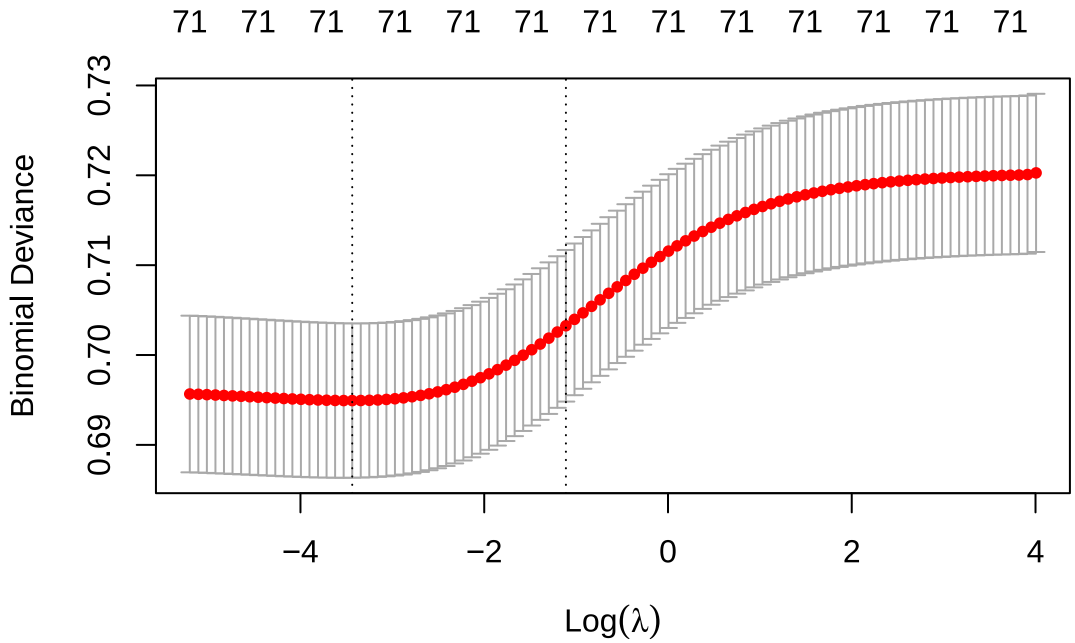
iii. Produce the trace plot based on `lambda.min`. Give interpretations.

```
plot_glmnet(lasso_fit, readmission_train, features_to_plot = 6, lambda = lasso_fit$lambda.min)
```



iv. Repeat steps i-iii, this time using ridge regression

```
set.seed(3)
ridge_fit = cv.glmnet(readmitted ~ ., family='binomial',
                      alpha = 0,
                      nfolds = 10,
                      data = readmission_train)
plot(ridge_fit)
```



```
plot_glmnet(ridge_fit, readmission_train, features_to_plot = 6, lambda = ridge_fit$lambda.min)
```

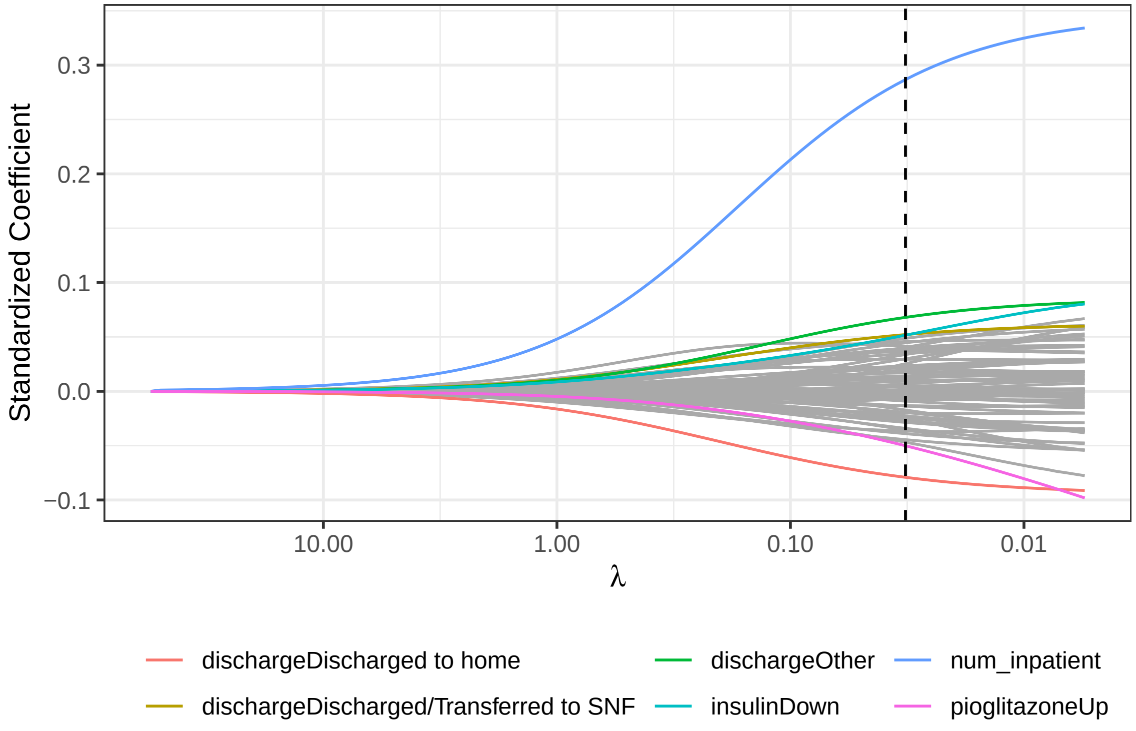


Table 3: Misclassification error of Lasso regression with different thresholds

Threshold	Lasso error	Lasso FPR	Lasso FNR
0.1	0.49	0.51	0.30
0.3	0.11	0.02	0.98
0.5	0.10	0.00	0.99
0.7	0.10	0.00	1.00

3 Model evaluation with different probability thresholds

i. Let us consider the following prediction rule. For a predicted $\hat{y}$, if it exceeds some `threshold`, we let the final prediction be one; otherwise, we let the final prediction be zero. For a grid of `threshold` = 0.1, 0.3, 0.5, 0.7, compute the mis-classification error, false positive rate (FP / N), and false negative rate (FN / P) for lasso regression using `lambda.min`. Which `threshold` is the best? Tabulate the results in a nice table.

```r
get_metrics = function(yhat, y){
  err = mean(yhat != y)
  positive = sum(y)
  negative = length(y) - positive
  fpr = sum((yhat == 1) & (y == 0)) / negative
  fnr = sum((yhat == 0) & (y == 1)) / positive
  return(c(err, fpr, fnr))
}
```

```r
groundtruth = readmission_test %>% select(readmitted) %>% pull()
lasso_predictions = predict(lasso_fit, newdata = readmission_test,
                            s = 'lambda.min', type='response') %>%
  as.numeric()

lasso_err = c(0, 0, 0, 0)
lasso_fpr = c(0, 0, 0, 0)
lasso_fnr = c(0, 0, 0, 0)
thresholds = c(0.1, 0.3, 0.5, 0.7)
for(i in 1:4) {
  lasso_predictions_binary = as.integer(lasso_predictions >= thresholds[i])
  curr_metrics = get_metrics(lasso_predictions_binary, groundtruth)
  lasso_err[i] = curr_metrics[1]
  lasso_fpr[i] = curr_metrics[2]
  lasso_fnr[i] = curr_metrics[3]
}

tibble(threshold = thresholds, error = lasso_err, fpr = lasso_fpr, fnr = lasso_fnr) %>%
    kable(format = "latex", row.names = NA,
        booktabs = TRUE, digits = 2,
        col.names = c("Threshold", "Lasso error", "Lasso FPR", "Lasso FNR"),
        caption = "Misclassification error of Lasso regression with different thresholds") %>%
    kable_styling(position = "center")
```

ii. Do the same thing for ridge. For the same level of threshold, which model is better?

Table 4: Misclassification error of Ridge regression with different thresholds

Threshold	Ridge error	Ridge FPR	Ridge FNR
0.1	0.50	0.53	0.30
0.3	0.11	0.01	0.97
0.5	0.10	0.00	1.00
0.7	0.10	0.00	1.00

```r
groundtruth = readmission_test %>% select(readmitted) %>% pull()
ridge_predictions = predict(ridge_fit, newdata = readmission_test,
                            s = 'lambda.min', type='response') %>%
  as.numeric()

ridge_err = c(0, 0, 0, 0)
ridge_fpr = c(0, 0, 0, 0)
ridge_fnr = c(0, 0, 0, 0)
thresholds = c(0.1, 0.3, 0.5, 0.7)
for(i in 1:4) {
  ridge_predictions_binary = as.integer(ridge_predictions >= thresholds[i])
  curr_metrics = get_metrics(ridge_predictions_binary, groundtruth)
  ridge_err[i] = curr_metrics[1]
  ridge_fpr[i] = curr_metrics[2]
  ridge_fnr[i] = curr_metrics[3]
}

tibble(threshold = thresholds, error = ridge_err, fpr = ridge_fpr, fnr = ridge_fnr) %>%
    kable(format = "latex", row.names = NA,
        booktabs = TRUE, digits = 2,
        col.names = c("Threshold", "Ridge error", "Ridge FPR", "Ridge FNR"),
        caption = "Misclassification error of Ridge regression with different thresholds") %>%
    kable_styling(position = "center")
```