Homework 4

Marshall Roll, Kenny Nhan, Emily Neuman

2022-11-01

```
## See answer to Q2 to explain irregularities

# library statements
library(ISLR)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(readr)
library(broom)
library(ggplot2)
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.0.0 --

## v dials        1.0.0     v tibble       3.1.8
## v infer        1.0.3     v tidyr        1.2.0
## v modeldata    1.0.0     v tune         1.0.0
## v parsnip      1.0.1     v workflows    1.0.0
## v purrr        0.3.4     v workflowsets 1.0.0
## v recipes      1.0.1     v yardstick    1.0.0
## v rsample      1.1.0

## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x purrr::discard()  masks scales::discard()
## x dplyr::filter()   masks stats::filter()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
library(probably)
```

```
##
## Attaching package: 'probably'

## The following objects are masked from 'package:base':
##
##     as.factor, as.ordered
```

```
tidymodels_prefer() # Resolves conflicts, prefers tidymodel functions
# read in data

stroke <- read_csv("https://raw.githubusercontent.com/MarshallRoll/STAT_253_Project/main/healthcare-data
```

```
## Rows: 5110 Columns: 12

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): gender, ever_married, work_type, Residence_type, bmi, smoking_status
## dbl (6): id, age, hypertension, heart_disease, avg_glucose_level, stroke
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# data cleaning
stroke_clean <- stroke %>%
  filter(bmi != "N/A") %>%
  mutate(bmi = as.numeric(bmi)) %>%
  filter(smoking_status != "Unknown") %>%
  select(-id) %>%
  mutate(stroke = relevel(factor(stroke), ref='0'))
```

Required Analysis

Start working on building a classification model to answer a research question on your data set. For HW4, only include your classification model work (leave your regression models work in another file).

For this homework,

Specify the research question for a classification task.

Try to implement at least 2 different classification methods to answer your research question.

Reflect on the information gained from these two methods and how you might justify this method to others.

Keep in mind that the final project will require you to complete the pieces below. Use this as a guide for your work but don't try to accomplish everything for HW4:

Answers to Questions:

We will create a model to effectively answer the question, will someone have a stroke.

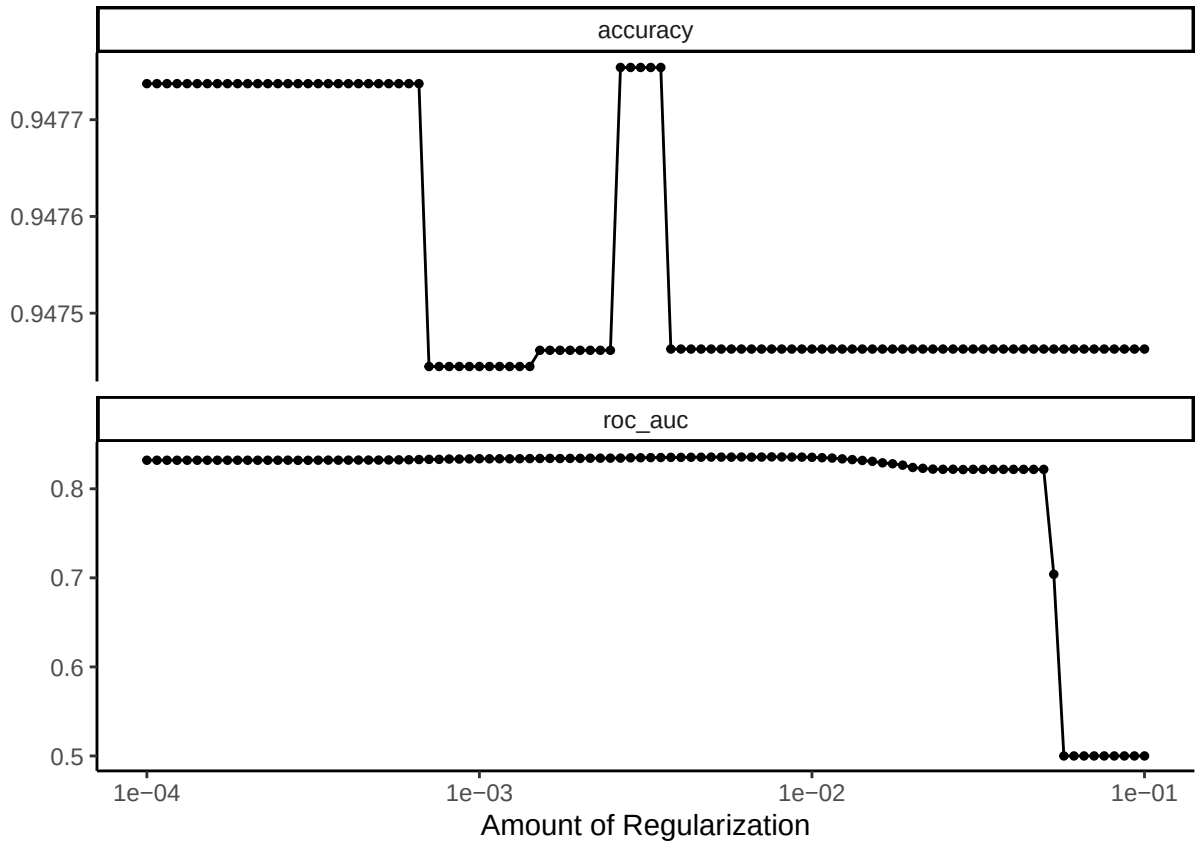
Our first classification method will be creating a logistic regression model using LASSO.

```
# creation of cv folds
stroke_cv <- vfold_cv(stroke_clean, v = 15)
```

```
# Logistic LASSO Regression Model Spec
logistic_lasso_spec_tune <- logistic_reg() %>%
    set_engine('glmnet') %>%
    set_args(mixture = 1, penalty = tune()) %>%
    set_mode('classification')

# Recipe
logistic_rec <- recipe(stroke~ ., data = stroke_clean) %>%
    step_normalize(all_numeric_predictors()) %>%
    step_dummy(all_nominal_predictors())

# Workflow (Recipe + Model)
log_lasso_wf <- workflow() %>%
    add_recipe(logistic_rec) %>%
    add_model(logistic_lasso_spec_tune)

# Tune Model (trying a variety of values of Lambda penalty)
penalty_grid <- grid_regular(
  penalty(range = c(-4, -1)), #log10 transformed  (kept moving min down from 0)
  levels = 100)

tune_output <- tune_grid(
  log_lasso_wf, # workflow
  resamples = stroke_cv, # cv folds
  metrics = metric_set(roc_auc,accuracy),
  control = control_resamples(save_pred = TRUE, event_level = 'second'),
  grid = penalty_grid # penalty grid defined above
)
```

```
## ! Fold11: preprocessor 1/1, model 1/1 (predictions): There are new levels in a factor: Other
```

```
autoplot(tune_output) + theme_classic()
```




```
# Select Penalty
best_se_penalty <- select_by_one_std_err(tune_output, metric = 'roc_auc', desc(penalty)) # choose penal
best_se_penalty
```

```
## # A tibble: 1 x 9
##   penalty .metric .estimator  mean     n std_err .config             .best .bound
##     <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>               <dbl>  <dbl>
## 1  0.0498 roc_auc binary     0.822    15  0.0166 Preprocessor1_Mod~ 0.836  0.818
```

```
# Fit Final Model
final_fit_se <- finalize_workflow(log_lasso_wf, best_se_penalty) %>% # incorporates penalty value to wo
    fit(data = stroke_clean)

final_fit_se %>% tidy()
```

```
## # A tibble: 16 x 3
##    term                         estimate penalty
##    <chr>                           <dbl>   <dbl>
##  1 (Intercept)                    -2.90    0.0498
##  2 age                             0.0874  0.0498
##  3 hypertension                    0       0.0498
##  4 heart_disease                   0       0.0498
##  5 avg_glucose_level               0       0.0498
##  6 bmi                             0       0.0498
##  7 gender_Male                     0       0.0498
```

```
##  8 gender_Other                  0       0.0498
##  9 ever_married_Yes              0       0.0498
## 10 work_type_Govt_job            0       0.0498
## 11 work_type_Never_worked        0       0.0498
## 12 work_type_Private             0       0.0498
## 13 work_type_Self.employed       0       0.0498
## 14 Residence_type_Urban          0       0.0498
## 15 smoking_status_never.smoked   0       0.0498
## 16 smoking_status_smokes         0       0.0498
```

```
glmnet_output <- final_fit_se %>% extract_fit_engine()

# Create a boolean matrix (predictors x lambdas) of variable exclusion
bool_predictor_exclude <- glmnet_output$beta==0

# Loop over each variable
var_imp <- sapply(seq_len(nrow(bool_predictor_exclude)), function(row) {
    # Extract coefficient path (sorted from highest to lowest lambda)
    this_coeff_path <- bool_predictor_exclude[row,]
    # Compute and return the # of lambdas until this variable is out forever
    ncol(bool_predictor_exclude) - which.min(this_coeff_path) + 1
})

# Create a dataset of this information and sort
var_imp_data <- tibble(
    var_name = rownames(bool_predictor_exclude),
    var_imp = var_imp
)
var_imp_data %>% arrange(desc(var_imp))
```

```
## # A tibble: 15 x 2
##    var_name                    var_imp
##    <chr>                         <dbl>
##  1 gender_Other                     58
##  2 work_type_Govt_job               58
##  3 work_type_Never_worked           58
##  4 Residence_type_Urban             58
##  5 age                              57
##  6 hypertension                     49
##  7 avg_glucose_level                49
##  8 heart_disease                    47
##  9 work_type_Private                34
## 10 smoking_status_smokes            32
## 11 work_type_Self.employed          26
## 12 smoking_status_never.smoked      24
## 13 bmi                              19
## 14 gender_Male                      19
## 15 ever_married_Yes                 19
```

```
# evalutation metrics

# CV results for "best lambda"
tune_output %>%
```

```
    collect_metrics() %>%
    filter(penalty == best_se_penalty %>% pull(penalty))
```

```
## # A tibble: 2 x 7
##   penalty .metric  .estimator  mean     n std_err .config
##     <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1  0.0498 accuracy binary     0.947    15 0.00495 Preprocessor1_Model090
## 2  0.0498 roc_auc  binary     0.822    15 0.0166  Preprocessor1_Model090
```

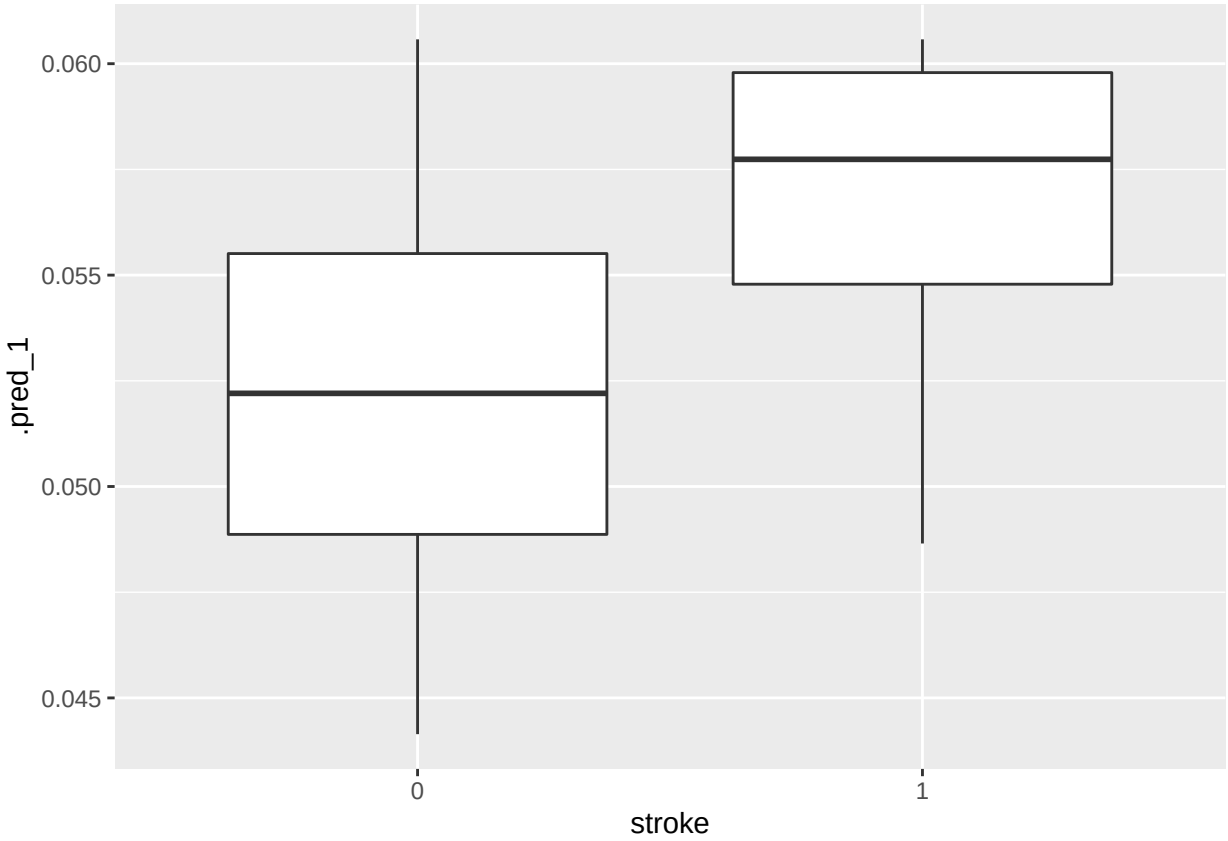
```
stroke_clean %>%
    count(stroke)
```

```
## # A tibble: 2 x 2
##   stroke     n
##   <fct>  <int>
## 1 0       3246
## 2 1        180
```

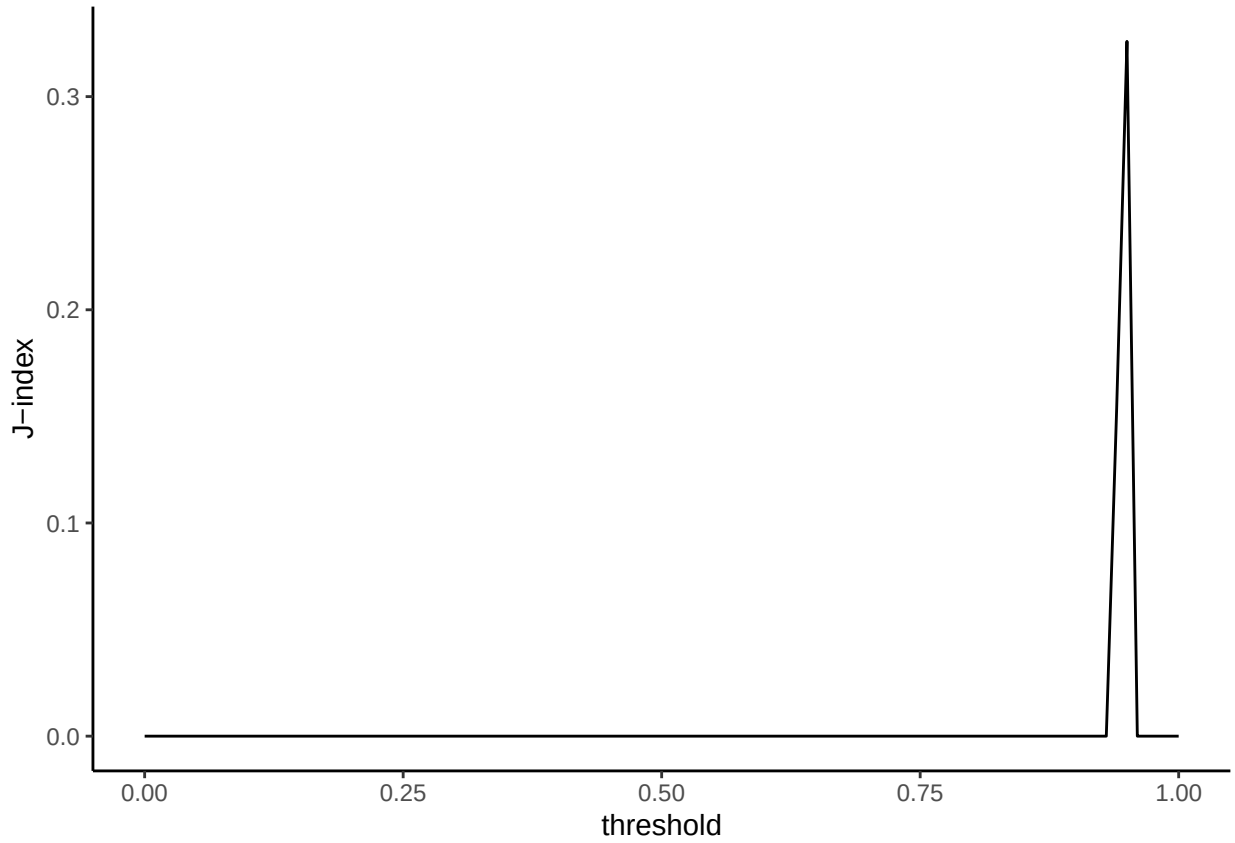
```
# Compute the NIR
3246/(3246+180)
```

```
## [1] 0.9474606
```

```
final_output <- final_fit_se %>% predict(new_data = stroke_clean, type='prob') %>% bind_cols(stroke_clea

final_output %>%
  ggplot(aes(x = stroke, y = .pred_1)) +
  geom_boxplot()
```




```
# thresholds in terms of reference level
threshold_output <- final_output %>%
    threshold_perf(truth = stroke, estimate = .pred_0, thresholds = seq(0,1,by=.01))

# J-index v. threshold for not_spam
threshold_output %>%
    filter(.metric == 'j_index') %>%
    ggplot(aes(x = .threshold, y = .estimate)) +
    geom_line() +
    labs(y = 'J-index', x = 'threshold') +
    theme_classic()
```




```
threshold_output %>%
    filter(.metric == 'j_index') %>%
    arrange(desc(.estimate))
```

```
## # A tibble: 101 x 4
##    .threshold .metric .estimator .estimate
##         <dbl> <chr>   <chr>          <dbl>
##  1       0.95 j_index binary         0.326
##  2       0.94 j_index binary         0.153
##  3       0    j_index binary         0
##  4       0.01 j_index binary         0
##  5       0.02 j_index binary         0
##  6       0.03 j_index binary         0
##  7       0.04 j_index binary         0
##  8       0.05 j_index binary         0
##  9       0.06 j_index binary         0
## 10       0.07 j_index binary         0
## # ... with 91 more rows
```

```
threshold_output %>%
    filter(.metric == 'distance') %>%
    arrange(.estimate)
```

```
## # A tibble: 101 x 4
```

```
##    .threshold .metric  .estimator .estimate
##         <dbl> <chr>    <chr>          <dbl>
##  1       0.95 distance binary         0.440
##  2       0.94 distance binary         0.659
##  3       0    distance binary         1
##  4       0.01 distance binary         1
##  5       0.02 distance binary         1
##  6       0.03 distance binary         1
##  7       0.04 distance binary         1
##  8       0.05 distance binary         1
##  9       0.06 distance binary         1
## 10       0.07 distance binary         1
## # ... with 91 more rows
```

```
log_metrics <- metric_set(accuracy,sens,yardstick::spec)

final_output %>%
    mutate(.pred_class = make_two_class_pred(.pred_0, levels(stroke), threshold = .91)) %>%
    log_metrics(truth = stroke, estimate = .pred_class, event_level = 'second')
```

```
## # A tibble: 3 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.947
## 2 sens     binary         0
## 3 spec     binary         1
```

We choose this threshold in an effort tor prioritize a higher specificity in the context of the data. When using this model to predict whether or not someone will have a stroke, it is important that the number of false positives is higher than the number of false negatives. Meaning that we are more likely to over predict a stroke than tell someone who is going to have a stroke they are not going to.

Classification - Methods Indicate at least 2 different methods used to answer your classification research question. Describe what you did to evaluate the models explored. Indicate how you estimated quantitative evaluation metrics. Describe the goals / purpose of the methods used in the overall context of your research investigations. Classification - Results Summarize your final model and justify your model choice (see below for ways to justify your choice). Compare the different classification models tried in light of evaluation metrics, variable importance, and data context. Display evaluation metrics for different models in a clean, organized way. This display should include both the estimated metric as well as its standard deviation. (This won't be available from OOB error estimation. If using OOB, don't worry about reporting the SD.) Broadly summarize conclusions from looking at these evaluation metrics and their measures of uncertainty.

Classification - Conclusions - Interpret evaluation metric(s) for the final model in context. Does the model show an acceptable amount of error? - If using OOB error estimation, display the test (OOB) confusion matrix, and use it to interpret the strengths and weaknesses of the final model. - Summarization should show evidence of acknowledging the data context in thinking about the sensibility of these results.