Biostat 203B Homework 5

Due Mar 20 @ 11:59PM

AUTHOR
Ningke Zhang 705834790

Predicting ICU duration

Using the ICU cohort `mimiciv_icu_cohort.rds` you built in Homework 4, develop at least three machine learning approaches (logistic regression with enet regularization, random forest, boosting, SVM, MLP, etc) plus a model stacking approach for predicting whether a patient's ICU stay will be longer than 2 days. You should use the `los_long` variable as the outcome. You algorithms can use patient demographic information (gender, age at ICU `intime`, marital status, race), ICU admission information (first care unit), the last lab measurements before the ICU stay, and first vital measurements during ICU stay as features. You are welcome to use any feature engineering techniques you think are appropriate; but make sure to not use features that are not available at an ICU stay's `intime`. For instance, `last_careunit` cannot be used in your algorithms.

1. Data preprocessing and feature engineering.

```
# load libraries
library(stacks)
library(tidymodels)
```

```
── Attaching packages ────────────────────────────────────── tidymodels 1.3.0 ──

✔ broom        1.0.7     ✔ recipes      1.1.1
✔ dials        1.4.0     ✔ rsample      1.2.1
✔ dplyr        1.1.4     ✔ tibble       3.2.1
✔ ggplot2      3.5.1     ✔ tidyr        1.3.1
✔ infer        1.0.7     ✔ tune         1.3.0
✔ modeldata    1.4.0     ✔ workflows    1.2.0
✔ parsnip      1.3.1     ✔ workflowsets 1.1.0
✔ purrr        1.0.4     ✔ yardstick    1.3.2

── Conflicts ───────────────────────────────────────── tidymodels_conflicts() ──
✖ purrr::discard() masks scales::discard()
✖ dplyr::filter()  masks stats::filter()
✖ dplyr::lag()     masks stats::lag()
✖ recipes::step()  masks stats::step()
```

```
library(dplyr)
library(recipes)
library(workflows)
library(tune)
library(glmnet)
```

```
Loading required package: Matrix


Attaching package: 'Matrix'

The following objects are masked from 'package:tidyr':

    expand, pack, unpack

Loaded glmnet 4.1-8
```

```
library(vip)
```

```

Attaching package: 'vip'

The following object is masked from 'package:utils':

    vi
```

```
library(ranger)
library(future)
library(xgboost)
```

```

Attaching package: 'xgboost'

The following object is masked from 'package:dplyr':

    slice
```

```
# read data
mimiciv_icu_cohort <- readRDS("../hw4/mimiciv_shiny/mimic_icu_cohort.rds") |>
  select(-c(intime,
            outtime,
            admittime,
            dischtime,
            deathtime,
            admit_provider_id,
            edregtime,
            edouttime,
            anchor_age,
            anchor_year,
            anchor_year_group,
            last_careunit,
            discharge_location,
            hospital_expire_flag,
            dod,
            los)
```

```
      ) |>
  mutate(los_long = as.factor(los_long)) |>
  print(width = Inf)
```

```
# A tibble: 94,458 × 26
   subject_id  hadm_id  stay_id first_careunit
        <int>    <int>    <int> <fct>
 1   10000032 29079034 39553978 Medical Intensive Care Unit (MICU)
 2   10000690 25860671 37081114 Medical Intensive Care Unit (MICU)
 3   10000980 26913865 39765666 Medical Intensive Care Unit (MICU)
 4   10001217 24597018 37067082 Surgical Intensive Care Unit (SICU)
 5   10001217 27703517 34592300 Surgical Intensive Care Unit (SICU)
 6   10001725 25563031 31205490 Medical/Surgical Intensive Care Unit (MICU/SICU)
 7   10001843 26133978 39698942 Medical/Surgical Intensive Care Unit (MICU/SICU)
 8   10001884 26184834 37510196 Medical Intensive Care Unit (MICU)
 9   10002013 23581541 39060235 Cardiac Vascular Intensive Care Unit (CVICU)
10   10002114 27793700 34672098 Other
   admission_type              admission_location     insurance language
   <fct>                       <fct>                  <chr>     <chr>
 1 EW EMER.                    EMERGENCY ROOM         Medicaid  English
 2 EW EMER.                    EMERGENCY ROOM         Medicare  English
 3 EW EMER.                    EMERGENCY ROOM         Medicare  English
 4 EW EMER.                    EMERGENCY ROOM         Private   Other
 5 Other                       PHYSICIAN REFERRAL     Private   Other
 6 EW EMER.                    Other                  Private   English
 7 URGENT                      TRANSFER FROM HOSPITAL Medicare  English
 8 OBSERVATION ADMIT           EMERGENCY ROOM         Medicare  English
 9 SURGICAL SAME DAY ADMISSION PHYSICIAN REFERRAL     Medicare  English
10 OBSERVATION ADMIT           PHYSICIAN REFERRAL     Medicaid  English
   marital_status race  gender intime_age hematocrit bicarbonate   wbc
   <chr>          <chr> <chr>       <int>      <dbl>       <dbl> <dbl>
 1 WIDOWED        WHITE F              52       41.1          25   6.9
 2 WIDOWED        WHITE F              86       36.1          26   7.1
 3 MARRIED        BLACK F              76       27.3          21   5.3
 4 MARRIED        WHITE F              55       38.1          22  15.7
 5 MARRIED        WHITE F              55       37.4          30   5.4
 6 MARRIED        WHITE F              46       NA            NA  NA
 7 SINGLE         WHITE M              76       31.4          28  10.4
 8 MARRIED        BLACK F              77       39.7          30  12.2
 9 SINGLE         Other F              57       34.9          24   7.2
10 <NA>           Other M              56       34.3          18  16.8
   creatinine chloride sodium potassium glucose respiratory_rate
        <dbl>    <dbl>  <dbl>     <dbl>   <dbl>            <dbl>
 1        0.7       95    126       6.7     102               24
 2        1        100    137       4.8      85               27
 3        2.3      109    144       3.9      89               24
 4        0.6      108    142       4.2     112               18
 5        0.5      104    142       4.1      87               17
 6       NA         98    139       4.1      NA               19
 7        1.3       97    138       3.9     131               17
```

```
 8        1.1       88    130       4.5     141               16
 9        0.9      102    137       3.5     288               14
10        3.1       NA    125       6.5      95               22
   non_invasive_blood_pressure_diastolic heart_rate temperature_fahrenheit
                                   <dbl>      <dbl>                  <dbl>
 1                                    48         91                   98.7
 2                                    63         80                   97.7
 3                                   127         77                   98
 4                                    90         86                   98.5
 5                                    97         96                   97.6
 6                                    56         86                   97.7
 7                                    85        131                   97.9
 8                                    49         60                   98.1
 9                                    70         80                   97.2
10                                    80        111                   97.9
   non_invasive_blood_pressure_systolic los_long
                                  <dbl> <fct>
 1                                   84 FALSE
 2                                  107 TRUE
 3                                  158 FALSE
 4                                  151 FALSE
 5                                  167 FALSE
 6                                   73 FALSE
 7                                  112 FALSE
 8                                  180 TRUE
 9                                  104 FALSE
10                                  112 TRUE
# ℹ 94,448 more rows
```

2. Partition data into 50% training set and 50% test set. Stratify partitioning according to `los_long`. For grading purpose, sort the data by `subject_id`, `hadm_id`, and `stay_id` and use the seed `203` for the initial data split. Below is the sample code.

```
set.seed(203)

mimiciv_icu_cohort <- mimiciv_icu_cohort |>
  arrange(subject_id, hadm_id, stay_id) |>
  select(-c(subject_id, hadm_id, stay_id))
mimiciv_icu_cohort <- mimiciv_icu_cohort |> drop_na()

data_split <- initial_split(mimiciv_icu_cohort,
                            strata = "los_long",
                            prop = 0.5)

icu_other <- training(data_split)
icu_test <- testing(data_split)
```

3. Train and tune the models using the training set.

- For logistic regression with elasticnet regularization, has 0.574 accuracy and 0.605 AUC in train data, the most important features are non invasive blood pressure systolic, frist care unit, and heart rate.
- For random forest, has 0.596 accuracy and 0.635 AUC in train data, the most important features are creatinine, intime age, and non invasive blood pressure systolic.
- For boosting, has 0.601 accuracy and 0.638 AUC in train data, the most important features are non invasive blood pressure systolic, intime age, and hematocrit.

```
# read models
logit_mod <- readRDS("final_fit_logistic_lastfit.rds")
rf_mod <- readRDS("final_fit_rf_lastfit.rds")
gb_mod <- readRDS("final_fit_gb_lastfit.rds")

logit_metrics <- logit_mod |> collect_metrics() |>
  filter(.metric %in% c("roc_auc", "accuracy"))
rf_metrics <- rf_mod |> collect_metrics() |>
  filter(.metric %in% c("roc_auc", "accuracy"))
gb_metrics <- gb_mod |> collect_metrics() |>
  filter(.metric %in% c("roc_auc", "accuracy"))

print(logit_metrics)
```

```
# A tibble: 2 × 4
  .metric  .estimator .estimate .config
  <chr>    <chr>          <dbl> <chr>
1 accuracy binary         0.574 Preprocessor1_Model1
2 roc_auc  binary         0.605 Preprocessor1_Model1
```

```
print(rf_metrics)
```

```
# A tibble: 2 × 4
  .metric  .estimator .estimate .config
  <chr>    <chr>          <dbl> <chr>
1 accuracy binary         0.596 Preprocessor1_Model1
2 roc_auc  binary         0.635 Preprocessor1_Model1
```

```
print(gb_metrics)
```

```
# A tibble: 2 × 4
  .metric  .estimator .estimate .config
  <chr>    <chr>          <dbl> <chr>
1 accuracy binary         0.601 Preprocessor1_Model1
2 roc_auc  binary         0.638 Preprocessor1_Model1
```

```
### Stacking model
recipe <-
  recipe(los_long ~ ., data = icu_other) |>
  step_impute_median(all_numeric_predictors()) |>
  step_impute_mode(all_nominal_predictors()) |>
```

```
  step_novel(all_nominal_predictors()) |>
  step_unknown(all_nominal_predictors()) |>
  step_dummy(all_nominal_predictors()) |>
  step_nzv(all_predictors()) |>
  step_normalize(all_numeric_predictors(), -all_outcomes())

folds <- vfold_cv(icu_other, v = 2, strata = los_long)

#Logistic regression with elasticnet regularization
logit_mod <- logistic_reg(penalty = tune(), mixture = tune()) |>
  set_engine("glmnet", standardize = FALSE) |>
  set_mode("classification")

logit_wf <- workflow() |>
  add_recipe(recipe) |>
  add_model(logit_mod)

param_grid <- grid_regular(
  penalty(range = c(-6, 2)),
  mixture(range = c(0, 1)),
  levels = 5
)

logit_stacked <- logit_wf |>
  tune_grid(
    resamples = folds,
    grid = param_grid,
    metrics = metric_set(roc_auc, accuracy),
    control = control_stack_grid()
  )
logit_stacked

#Random Forest
rf_mod <-
  rand_forest(
    mode = "classification",
    mtry = tune(),
    trees = tune(),
    min_n = tune()) |>
  set_engine("ranger", importance = "permutation")

rf_wf <- workflow() |>
  add_recipe(recipe) |>
  add_model(rf_mod)

rf_grid <- grid_regular(
  mtry(range = c(2, 6)),
  trees(range = c(150, 200)),
  min_n(range = c(5, 10)),
  levels = 3
)
```

```
rf_stacked <- rf_wf |>
  tune_grid(
    resamples = folds,
    grid = rf_grid,
    metrics = metric_set(roc_auc, accuracy),
    control = control_stack_grid()
  )
rf_stacked

#Boosting
gb_mod <-
  boost_tree(
    mode = "classification",
    trees = 600,
    tree_depth = tune(),
    learn_rate = tune()
  ) |>
  set_engine("xgboost")

gb_wf <- workflow() |>
  add_recipe(recipe) |>
  add_model(gb_mod)

gb_grid <- grid_regular(
  tree_depth(range = c(3L, 8L)),
  learn_rate(range = c(-3, -0.5), trans = log10_trans()),
  levels = 5
)

gb_stacked <- gb_wf |>
  tune_grid(
    resamples = folds,
    grid = gb_grid,
    metrics = metric_set(roc_auc, accuracy),
    control = control_stack_grid()
  )
gb_stacked
```

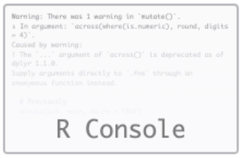
```
class(logit_stacked)
class(rf_stacked)
class(gb_stacked)

# define the stacking model
stacked_model <- stacks() |>
  add_candidates(logit_stacked) |>
  add_candidates(rf_stacked) |>
  add_candidates(gb_stacked)

stacked_model <- stacked_model |>
```

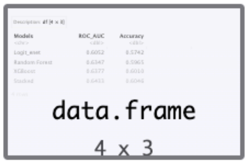
```
  blend_predictions(penalty = 1e-4, metric = metric_set(roc_auc, accuracy)) |>
  fit_members()


#plot the stacked model
autoplot(stacked_model)

# compute the performance of the stacked model
#auc
stacked_results_prob <- stacked_model |>
  predict(new_data = icu_test, type = "prob")
stacked_results_prob
stack_auc <- stacked_results_prob |>
  bind_cols(icu_test) |>
  roc_auc(truth = los_long, .pred_TRUE, event_level = "second")
stack_auc
#accuracy
stacked_results_class <- stacked_model |>
  predict(new_data = icu_test, type = "class")
stack_acc <- stacked_results_class |>
  bind_cols(icu_test) |>
  accuracy(truth = los_long, estimate = .pred_class)
stack_acc
```

4. Compare model classification performance on the test set. Report both the area under ROC curve and accuracy for each machine learning algorithm and the model stacking. Interpret the results. What are the most important features in predicting long ICU stays? How do the models compare in terms of performance and interpretability?

```
# Summary of the performance

logit_auc <- logit_metrics |> filter(.metric == "roc_auc") |> pull(.estimate)
rf_auc <- rf_metrics |> filter(.metric == "roc_auc") |> pull(.estimate)
gb_auc <- gb_metrics |> filter(.metric == "roc_auc") |> pull(.estimate)
stack_auc <- stack_auc$.estimate

logit_acc <- logit_metrics |> filter(.metric == "accuracy") |> pull(.estimate)
rf_acc <- rf_metrics |> filter(.metric == "accuracy") |> pull(.estimate)
gb_acc <- gb_metrics |> filter(.metric == "accuracy") |> pull(.estimate)
stack_acc <- stack_acc$.estimate

Models <- c("Logit_enet", "Random Forest", "XGBoost", "Stacked")
ROC_AUC <- c(logit_auc, rf_auc, gb_auc, stack_auc)
Accuracy <- c(logit_acc, rf_acc, gb_acc, stack_acc)

model_performance <- data.frame(Models, ROC_AUC, Accuracy) |>
  mutate(across(where(is.numeric), round, digits = 4)) |>
  print()
```

Summary:





Description: df [4 × 3]

Models <chr>	ROC_AUC <dbl>	Accuracy <dbl>
Logit_enet	0.6052	0.5742
Random Forest	0.6347	0.5965
XGBoost	0.6377	0.6010
Stacked	0.6433	0.6046

4 rows

According to the prediction results of the four models, Stacking achieved the best performance (AUC = 0.6433, Accuracy = 60.46%), indicating that combining multiple models effectively enhances prediction accuracy.
Logistic Regression (AUC = 0.6052) had the lowest performance, suggesting that linear relationships alone may not be sufficient to capture the complex patterns of ICU stay duration.
XGBoost (~0.638 AUC) slightly outperformed Random Forest (~0.635 AUC), implying that boosting-based approaches are more effective in this dataset.
So our most important features will follow the vip results of XGBoost: non invasive blood pressure systolic, intime age, and hematocrit.