Final Model - readmission_180d

Eduardo Yuki Yada

Global parameters

```
k <- params$k # Number of folds for cross validation
grid_size <- params$grid_size # Number of parameter combination to tune on each model
max_auc_loss <- params$max_auc_loss # Max accepted loss of AUC for reducing num of features
repeats <- params$repeats
Hmisc::list.tree(params)
```

```
##  params = list 5 (968 bytes)
## .  max_auc_loss = double 1= 0.01
## .  outcome_column = character 1= readmission_180d
## .  k = double 1= 10
## .  grid_size = double 1= 50
## .  repeats = double 1= 2
```

Imports

```
library(tidyverse)
library(yaml)
library(tidymodels)
library(usemodels)
library(vip)
library(kableExtra)
library(SHAPforxgboost)
library(xgboost)
library(Matrix)
library(mltools)
library(bonsai)
library(lightgbm)
library(pROC)
library(caret)
library(themis)

source("aux_functions.R")

select <- dplyr::select
```

Loading data

```
load('dataset/processed_data.RData')
load('dataset/processed_dictionary.RData')

columns_list <- yaml.load_file("./auxiliar/columns_list.yaml")

outcome_column <- params$outcome_column
features_list <- params$features_list

df[columns_list$outcome_columns] <- lapply(df[columns_list$outcome_columns], factor)
```

```
df <- mutate(df, across(where(is.character), as.factor))
```

```
dir.create(file.path("./auxiliar/final_model/hyperparameters/"),
           showWarnings = FALSE,
           recursive = TRUE)

dir.create(file.path("./auxiliar/final_model/performance/"),
           showWarnings = FALSE,
           recursive = TRUE)

dir.create(file.path("./auxiliar/final_model/selected_features/"),
           showWarnings = FALSE,
           recursive = TRUE)
```

Eligible features

```
cat_features_list = readRDS(sprintf(
  "./auxiliar/significant_columns/categorical_%s.rds",
  outcome_column
))

num_features_list = readRDS(sprintf(
  "./auxiliar/significant_columns/numerical_%s.rds",
  outcome_column
))

features_list = c(cat_features_list, num_features_list)
```

```
eligible_columns <- df_names %>%
  filter(momento.aquisicao == 'Admissão t0') %>%
  .$variable.name

exception_columns <- c('death_intraop', 'death_intraop_1', 'disch_outcomes_t0')

correlated_columns = c('year_procedure_1', # com year_adm_t0
                       'age_surgery_1', # com age
                       'admission_t0', # com admission_pre_t0_count
                       'atb', # com meds_antimicrobianos
                       'classe_meds_cardio_qtde', # com classe_meds_qtde
                       'suporte_hemod', # com proced_invasivos_qtde,
                       'radiografia', # com exames_imagem_qtde
                       'ecg' # com metodos_graficos_qtde
                       )

eligible_features <- eligible_columns %>%
  base::intersect(c(columns_list$categorical_columns, columns_list$numerical_columns)) %>%
  setdiff(c(exception_columns, correlated_columns))

features = base::intersect(eligible_features, features_list)
```

Starting features:

```
gluedown::md_order(features, seq = TRUE, pad = TRUE)
```

1. sex
2. age
3. education_level
4. patient_state
5. underlying_heart_disease
6. heart_disease
7. nyha_basal

8. prior_mi
9. heart_failure
10. af
11. cardiac_arrest
12. transplant
13. valvopathy
14. endocardites
15. diabetes
16. renal_failure
17. hemodialysis
18. copd
19. comorbidities_count
20. procedure_type_1
21. reop_type_1
22. procedure_type_new
23. cied_final_1
24. cied_final_group_1
25. admission_pre_t0_count
26. admission_pre_t0_180d
27. icu_t0
28. dialysis_t0
29. n_procedure_t0
30. admission_t0_emergency
31. aco
32. antiarritmico
33. betabloqueador
34. ieca_bra
35. dva
36. digoxina
37. estatina
38. diuretico
39. vasodilatador
40. insuf_cardiaca
41. espironolactona
42. bloq_calcio
43. antiplaquetario_ev
44. insulina
45. anticonvulsivante
46. psicofarmacos
47. antifungico
48. antiviral
49. antiretroviral
50. classe_meds_qtde
51. meds_cardiovasc_qtde
52. meds_antimicrobianos
53. vni
54. ventilacao_mecanica
55. cec
56. transplante_cardiaco
57. cir_toracica
58. outros_proced_cirurgicos
59. icp
60. intervencao_cv
61. angioplastia
62. cateterismo
63. eletrofisiologia
64. cateter_venoso_central
65. proced_invasivos_qtde
66. cve_desf
67. transfusao
68. interconsulta

69. equipe_multiprof
70. holter
71. teste_esforco
72. espiro_ergoespiro
73. tilt_teste
74. metodos_graficos_qtde
75. laboratorio
76. cultura
77. analises_clinicas_qtde
78. citologia
79. biopsia
80. histopatologia_qtde
81. angio_rm
82. angio_tc
83. arteriografia
84. cintilografia
85. ecocardiograma
86. endoscopia
87. flebografia
88. pet_ct
89. ultrassom
90. tomografia
91. ressonancia
92. exames_imagem_qtde
93. bic
94. mpp
95. hospital_stay

Train test split (70%/30%)

```
set.seed(42)

if (outcome_column == 'readmission_30d') {
  df_split <- readRDS("dataset/split_object.rds")
} else {
  df_split <- initial_split(df, prop = .7, strata = all_of(outcome_column))
}

df_train <- training(df_split) %>% select(all_of(c(features, outcome_column)))
df_test <- testing(df_split) %>% select(all_of(c(features, outcome_column)))

df_folds <- vfold_cv(df_train, v = k,
                     strata = all_of(outcome_column),
                     repeats = repeats)
```

Feature Selection

```
custom_dummy_names <- function(var, lvl, ordinal = FALSE) {
  dummy_names(var, lvl, ordinal = FALSE, sep = "__")
}

model_fit_wf <- function(df_train, features, outcome_column, hyperparameters){
  model_recipe <-
    recipe(formula = sprintf("%s ~ .", outcome_column) %>% as.formula,
           data = df_train %>% select(all_of(c(features, outcome_column)))) %>%
    step_novel(all_nominal_predictors()) %>%
    step_unknown(all_nominal_predictors()) %>%
    step_other(all_nominal_predictors(), threshold = 0.05, other = ".merged") %>%
    step_dummy(all_nominal_predictors(), naming = custom_dummy_names)
```

```
  model_spec <-
    do.call(boost_tree, hyperparameters) %>%
    set_engine("lightgbm") %>%
    set_mode("classification")

  model_workflow <-
    workflow() %>%
    add_recipe(model_recipe) %>%
    add_model(model_spec)

  model_fit_rs <- model_workflow %>%
    fit_resamples(df_folds)

  model_fit <- model_workflow %>%
    fit(df_train)

  model_auc <- validation(model_fit, df_test, plot = F)

  raw_model <- parsnip::extract_fit_engine(model_fit)

  feature_importance <- lgb.importance(raw_model, percentage = TRUE) %>%
    separate(Feature, c("Feature","value"), "__", fill = 'right') %>%
    group_by(Feature) %>%
    summarise(Gain = sum(Gain),
              Cover = sum(Cover),
              Frequency = sum(Frequency)) %>%
    ungroup() %>%
    arrange(desc(Gain))

  cv_results <- collect_metrics(model_fit_rs) %>% filter(.metric == 'roc_auc')

  return(
    list(
      cv_auc =  cv_results$mean,
      cv_auc_std_err = cv_results$std_err,
      importance = feature_importance,
      auc = as.numeric(model_auc$auc),
      auc_lower = model_auc$ci[1],
      auc_upper = model_auc$ci[3]
    )
  )
}
```

```
hyperparameters <- readRDS(
  sprintf(
    "./auxiliar/model_selection/hyperparameters/lightgbm_%s.rds",
    outcome_column
  )
)

hyperparameters$sample_size <- 1

full_model <- model_fit_wf(df_train, features, outcome_column, hyperparameters)

sprintf('Full Model CV Train AUC: %.3f' ,full_model$cv_auc)
```

```
## [1] "Full Model CV Train AUC: 0.716"
```

```
sprintf('Full Model Test AUC: %.3f' ,full_model$auc)
```

```
## [1] "Full Model Test AUC: 0.693"
```

Features with zero importance on the initial model:

```
unimportant_features <- setdiff(features, full_model$importance$Feature)

unimportant_features %>%
  gluedown::md_order()
```

1. prior_mi
2. heart_failure
3. cardiac_arrest
4. transplant
5. valvopathy
6. endocardites
7. diabetes
8. renal_failure
9. hemodialysis
10. copd
11. reop_type_1
12. dialysis_t0
13. n_procedure_t0
14. admission_t0_emergency
15. aco
16. insuf_cardiaca
17. antiplaquetario_ev
18. insulina
19. anticonvulsivante
20. antifungico
21. antiretroviral
22. vni
23. ventilacao_mecanica
24. cec
25. transplante_cardiaco
26. cir_toracica
27. icp
28. intervencao_cv
29. angioplastia
30. cateterismo
31. eletrofisiologia
32. cateter_venoso_central
33. cve_desf
34. transfusao
35. interconsulta
36. teste_esforco
37. espiro_ergoespiro
38. tilt_teste
39. cultura
40. analises_clinicas_qtde
41. citologia
42. biopsia
43. angio_rm
44. angio_tc
45. arteriografia
46. cintilografia
47. pet_ct
48. ultrassom
49. tomografia
50. ressonancia
51. bic
52. mpp

```
trimmed_features <- full_model$importance$Feature
trimmed_model <- model_fit_wf(df_train, trimmed_features,
```

```
                                outcome_column, hyperparameters)

sprintf('Trimmed Model CV Train AUC: %.3f' ,trimmed_model$cv_auc)
```

```
## [1] "Trimmed Model CV Train AUC: 0.716"
```

```
sprintf('Trimmed Model Test AUC: %.3f' ,trimmed_model$auc)
```

```
## [1] "Trimmed Model Test AUC: 0.693"
```

```
selection_results <- tibble::tribble(
  ~`Tested Feature`, ~`Dropped`, ~`Number of Features`, ~`CV AUC`, ~`CV AUC Std Error`, ~`Total AUC Loss`, ~`Ins
  'None', TRUE, length(features), full_model$cv_auc, full_model$cv_auc_std_err, 0, 0
)

whitelist <- c()

if (full_model$cv_auc - trimmed_model$cv_auc < max_auc_loss) {
  current_features <- trimmed_features
  current_model <- trimmed_model
  current_auc_loss <- full_model$cv_auc - current_model$cv_auc
  instant_auc_loss <- full_model$cv_auc - current_model$cv_auc

  selection_results <- selection_results %>%
    add_row(`Tested Feature` = 'All unimportant',
            `Dropped` = TRUE,
            `Number of Features` = length(trimmed_features),
            `CV AUC` = current_model$cv_auc,
            `CV AUC Std Error` = current_model$cv_auc_std_err,
            `Total AUC Loss` = current_auc_loss,
            `Instant AUC Loss` = instant_auc_loss)
} else {
  current_features <- features
  current_model <- full_model
  current_auc_loss <- 0
}

while (current_auc_loss < max_auc_loss & mean(current_features %in% whitelist) < 1) {
  zero_importance_features <-
    setdiff(current_features, current_model$importance$Feature) %>%
    setdiff(whitelist)
  if (length(zero_importance_features) > 0) {
    current_least_important <- zero_importance_features[1]
  } else {
    current_least_important <-
      tail(setdiff(current_model$importance$Feature, whitelist), 1)
  }
  test_features <-
    setdiff(current_features, current_least_important)
  current_model <-
    model_fit_wf(df_train, test_features, outcome_column, hyperparameters)
  instant_auc_loss <-
    tail(selection_results %>% filter(Dropped) %>% .$`CV AUC`, n = 1) - current_model$cv_auc

  if (instant_auc_loss < max_auc_loss / 5 &
      current_auc_loss < max_auc_loss) {
    dropped <- TRUE
    current_features <- test_features
    current_auc_loss <- full_model$cv_auc - current_model$cv_auc
  } else {
    dropped <- FALSE
    whitelist <- c(whitelist, current_least_important)
```

```
  }

  selection_results <- selection_results %>%
    add_row(
      `Tested Feature` = current_least_important,
      `Dropped` = dropped,
      `Number of Features` = length(test_features),
      `CV AUC` = current_model$cv_auc,
      `CV AUC Std Error` = current_model$cv_auc_std_err,
      `Total AUC Loss` = current_auc_loss,
      `Instant AUC Loss` = instant_auc_loss
    )

  print(c(
    length(current_features),
    round(current_auc_loss, 4),
    round(instant_auc_loss, 4),
    current_least_important
  ))
}
```

```
## [1] "42"     "-1e-04" "1e-04"  "holter"
## [1] "41"     "-4e-04" "-3e-04" "af"
## [1] "40"       "-2e-04"   "2e-04"    "estatina"
## [1] "39"            "-2e-04"        "1e-04"         "heart_disease"
## [1] "38"                        "-2e-04"                  "0"
## [4] "underlying_heart_disease"
## [1] "37"         "-2e-04"     "-1e-04"     "nyha_basal"
## [1] "36"          "-4e-04"      "-2e-04"      "laboratorio"
## [1] "35"          "-5e-04"      "0"           "flebografia"
## [1] "34"                "-6e-04"            "-1e-04"             "exames_imagem_qtde"
## [1] "33"                   "-8e-04"                "-2e-04"                "proced_invasivos_qtde"
## [1] "32"               "-6e-04"           "1e-04"           "equipe_multiprof"
## [1] "31"                       "-7e-04"                  "-1e-04"
## [4] "outros_proced_cirurgicos"
## [1] "30"                 "-6e-04"            "1e-04"              "cied_final_group_1"
## [1] "29"       "-5e-04"   "0"        "ieca_bra"
## [1] "28"       "-2e-04"   "3e-04"    "digoxina"
## [1] "27"            "-9e-04"        "-7e-04"        "psicofarmacos"
## [1] "26"             "-0.0011"        "-2e-04"         "betabloqueador"
## [1] "25"          "-0.0014"     "-2e-04"      "bloq_calcio"
## [1] "24"         "-0.0015"    "-1e-04"     "endoscopia"
## [1] "23"             "-0.0015"        "0"              "ecocardiograma"
## [1] "22"        "-0.0014"   "2e-04"     "antiviral"
## [1] "21"      "-0.0012" "2e-04"   "sex"
## [1] "20"                  "-0.0012"             "0"                   "comorbidities_count"
## [1] "19"      "-0.0016" "-4e-04"  "dva"
## [1] "18"                   "-7e-04"              "9e-04"               "meds_antimicrobianos"
## [1] "17"                   "-3e-04"                "4e-04"                "metodos_graficos_qtde"
## [1] "16"                "-9e-04"             "-6e-04"             "procedure_type_new"
## [1] "15"            "-2e-04"        "7e-04"         "patient_state"
## [1] "14"        "0"         "2e-04"     "diuretico"
## [1] "13"              "0"               "0"               "espironolactona"
## [1] "12"           "9e-04"        "9e-04"        "cied_final_1"
## [1] "11"     "0.0017" "8e-04"  "icu_t0"
## [1] "10"              "0.003"           "0.0012"          "education_level"
## [1] "10"               "0.003"            "0.0026"           "procedure_type_1"
## [1] "9"             "0.0041"        "0.0011"        "vasodilatador"
## [1] "8"                     "0.0053"                "0.0012"               "admission_pre_t0_180d"
## [1] "7"                   "0.0057"             "5e-04"               "histopatologia_qtde"
## [1] "6"      "0.0055" "-2e-04" "age"
```

```
## [1] "5"                    "0.006"               "5e-04"               "meds_cardiovasc_qtde"
## [1] "5"             "0.006"         "0.0022"        "antiarritmico"
## [1] "5"                "0.006"            "0.0041"           "classe_meds_qtde"
## [1] "5"                     "0.006"                   "0.0115"                 "admission_pre_t0_count"
## [1] "5"             "0.006"         "0.0184"        "hospital_stay"
```

```
selection_results %>%
  rename(Features = `Number of Features`) %>%
  niceFormatting(digits = 4, label = 1)
```

Table 1:

Tested Feature	Dropped	Features	CV AUC	CV AUC Std Error	Total AUC Loss	Instant AUC Loss
None	TRUE	95	0.7156	0.0056	0.0000	0.0000
All unimportant	TRUE	43	0.7157	0.0057	-0.0001	-0.0001
holter	TRUE	42	0.7156	0.0057	-0.0001	0.0001
af	TRUE	41	0.7160	0.0057	-0.0004	-0.0003
estatina	TRUE	40	0.7158	0.0056	-0.0002	0.0002
heart_disease	TRUE	39	0.7157	0.0056	-0.0002	0.0001
underlying_heart_disease	TRUE	38	0.7157	0.0057	-0.0002	0.0000
nyha_basal	TRUE	37	0.7158	0.0057	-0.0002	-0.0001
laboratorio	TRUE	36	0.7160	0.0057	-0.0004	-0.0002
flebografia	TRUE	35	0.7160	0.0056	-0.0005	0.0000
exames_imagem_qtde	TRUE	34	0.7162	0.0057	-0.0006	-0.0001
proced_invasivos_qtde	TRUE	33	0.7163	0.0057	-0.0008	-0.0002
equipe_multiprof	TRUE	32	0.7162	0.0058	-0.0006	0.0001
outros_proced_cirurgicos	TRUE	31	0.7163	0.0057	-0.0007	-0.0001
cied_final_group_1	TRUE	30	0.7161	0.0057	-0.0006	0.0001
ieca_bra	TRUE	29	0.7161	0.0057	-0.0005	0.0000
digoxina	TRUE	28	0.7158	0.0058	-0.0002	0.0003
psicofarmacos	TRUE	27	0.7165	0.0057	-0.0009	-0.0007
betabloqueador	TRUE	26	0.7167	0.0058	-0.0011	-0.0002
bloq_calcio	TRUE	25	0.7169	0.0058	-0.0014	-0.0002
endoscopia	TRUE	24	0.7170	0.0057	-0.0015	-0.0001
ecocardiograma	TRUE	23	0.7171	0.0057	-0.0015	0.0000
antiviral	TRUE	22	0.7169	0.0057	-0.0014	0.0002
sex	TRUE	21	0.7167	0.0057	-0.0012	0.0002
comorbidities_count	TRUE	20	0.7167	0.0056	-0.0012	0.0000
dva	TRUE	19	0.7172	0.0057	-0.0016	-0.0004
meds_antimicrobianos	TRUE	18	0.7162	0.0057	-0.0007	0.0009
metodos_graficos_qtde	TRUE	17	0.7158	0.0057	-0.0003	0.0004
procedure_type_new	TRUE	16	0.7165	0.0057	-0.0009	-0.0006
patient_state	TRUE	15	0.7158	0.0057	-0.0002	0.0007
diuretico	TRUE	14	0.7155	0.0057	0.0000	0.0002
espironolactona	TRUE	13	0.7156	0.0059	0.0000	0.0000
cied_final_1	TRUE	12	0.7146	0.0058	0.0009	0.0009
icu_t0	TRUE	11	0.7138	0.0056	0.0017	0.0008
education_level	TRUE	10	0.7126	0.0058	0.0030	0.0012
procedure_type_1	FALSE	9	0.7100	0.0060	0.0030	0.0026
vasodilatador	TRUE	9	0.7115	0.0057	0.0041	0.0011
admission_pre_t0_180d	TRUE	8	0.7103	0.0058	0.0053	0.0012
histopatologia_qtde	TRUE	7	0.7098	0.0057	0.0057	0.0005
age	TRUE	6	0.7100	0.0056	0.0055	-0.0002
meds_cardiovasc_qtde	TRUE	5	0.7095	0.0056	0.0060	0.0005
antiarritmico	FALSE	4	0.7074	0.0057	0.0060	0.0022
classe_meds_qtde	FALSE	4	0.7054	0.0064	0.0060	0.0041
admission_pre_t0_count	FALSE	4	0.6980	0.0056	0.0060	0.0115
hospital_stay	FALSE	4	0.6912	0.0049	0.0060	0.0184

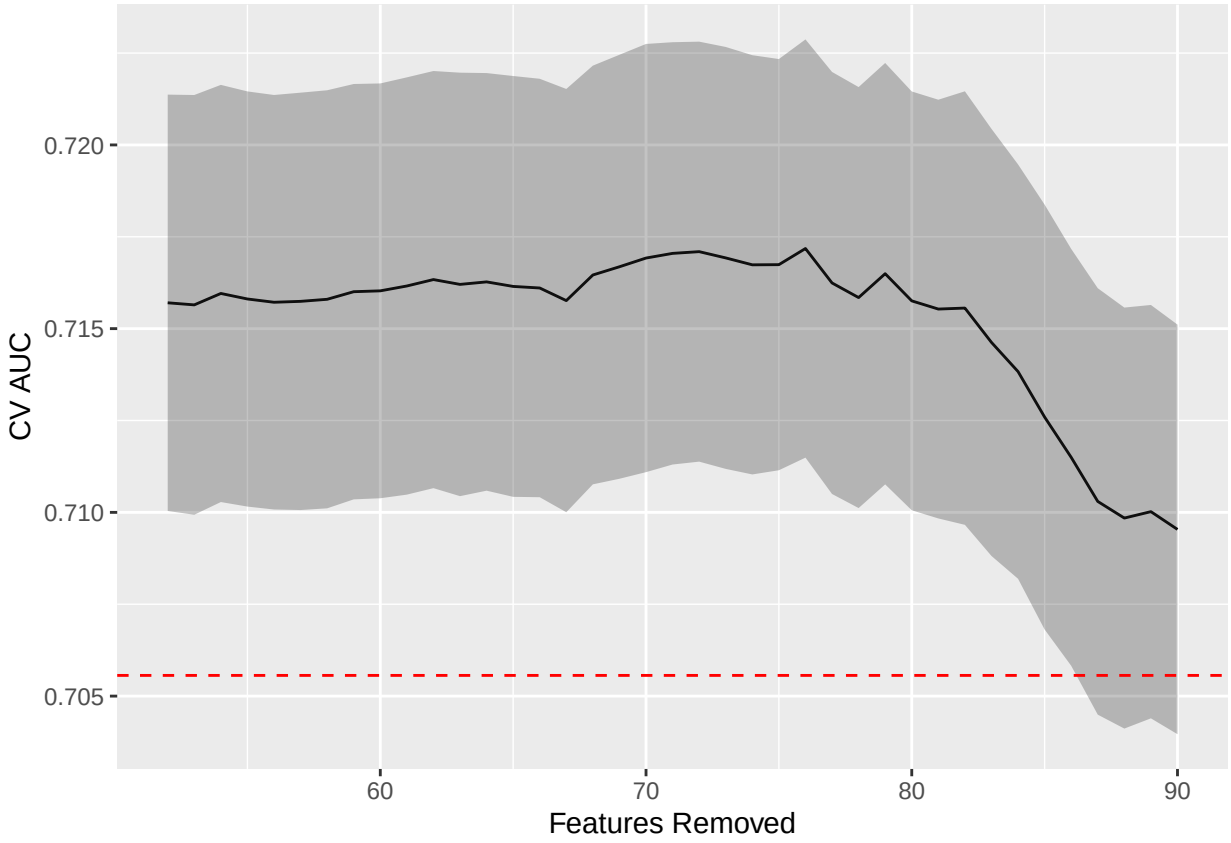
```
selected_features <- current_features

con <- file(sprintf('./auxiliar/final_model/selected_features/%s.yaml', outcome_column), "w")
write_yaml(selected_features, con)
close(con)

feature_selected_model <- model_fit_wf(df_train, selected_features,
                                       outcome_column, hyperparameters)

sprintf('Selected Model CV Train AUC: %.3f', feature_selected_model$cv_auc)
```

```
## [1] "Selected Model CV Train AUC: 0.710"
```

```
sprintf('Selected Model Test AUC: %.3f', feature_selected_model$auc)
```

```
## [1] "Selected Model Test AUC: 0.686"
```

```
 selection_results <- selection_results %>%
  filter(`Number of Features` < length(features)) %>%
  mutate(`Features Removed` = length(features) - `Number of Features`,
         `CV AUC Low` = `CV AUC` - `CV AUC Std Error`,
         `CV AUC High` = `CV AUC` + `CV AUC Std Error`)

selection_results %>%
  filter(Dropped) %>%
  ggplot(aes(x = `Features Removed`, y = `CV AUC`,
             ymin = `CV AUC Low`, ymax = `CV AUC High`)) +
  geom_line() +
  geom_ribbon(alpha = .3) +
  geom_hline(yintercept = full_model$cv_auc - max_auc_loss,
             linetype = "dashed", color = "red")
```



Hyperparameter tuning

Selected features:

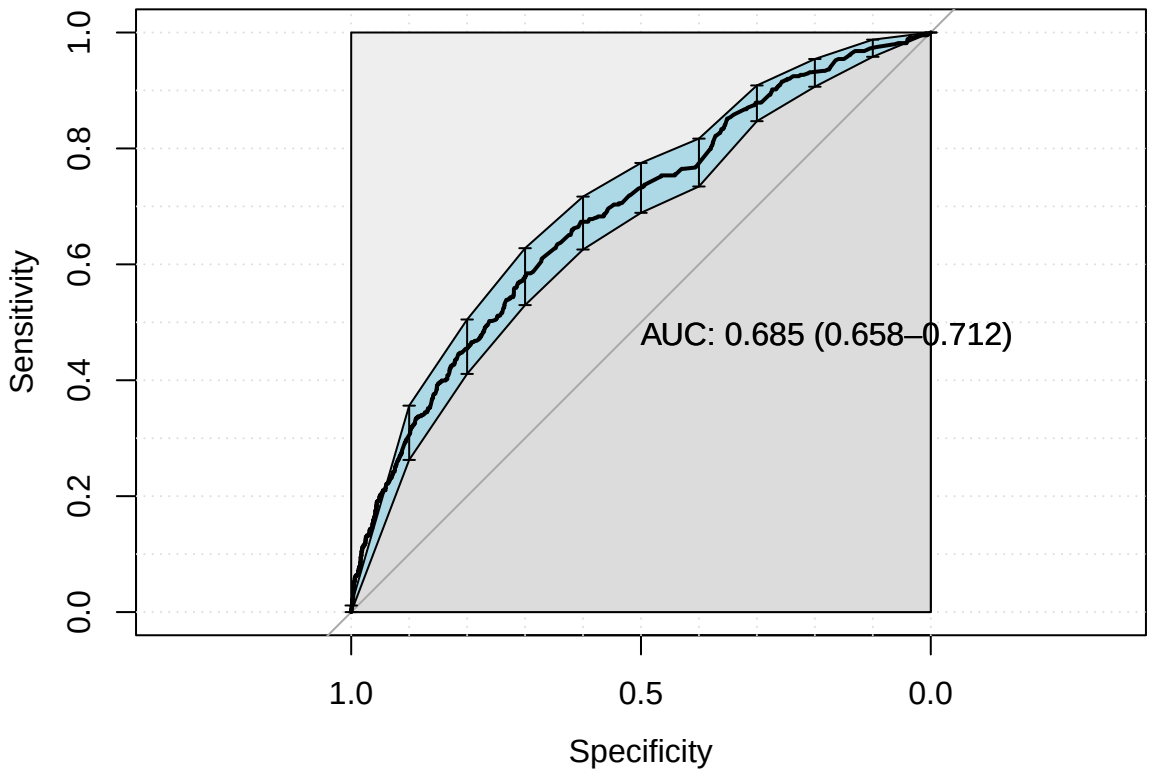
```
gluedown::md_order(selected_features, seq = TRUE, pad = TRUE)
```

1. hospital_stay
2. admission_pre_t0_count
3. classe_meds_qtde
4. antiarritmico
5. procedure_type_1

Standard

```
lightgbm_recipe <-
  recipe(formula = sprintf("%s ~ .", outcome_column) %>% as.formula,
         data = df_train %>% select(all_of(c(selected_features, outcome_column)))) %>%
  step_novel(all_nominal_predictors()) %>%
  step_unknown(all_nominal_predictors()) %>%
  step_other(all_nominal_predictors(), threshold = 0.05, other = ".merged") %>%
  step_dummy(all_nominal_predictors())

lightgbm_tuning <- function(recipe) {

  lightgbm_spec <- boost_tree(
    trees = tune(),
    min_n = tune(),
    tree_depth = tune(),
    learn_rate = tune(),
    sample_size = 1.0
  ) %>%
    set_engine("lightgbm",
               nthread = 8) %>%
    set_mode("classification")

  lightgbm_grid <- grid_latin_hypercube(
    trees(range = c(25L, 150L)),
    min_n(range = c(2L, 100L)),
    tree_depth(range = c(2L, 15L)),
    learn_rate(range = c(-3, -1), trans = log10_trans()),
    size = grid_size
  )

  lightgbm_workflow <-
    workflow() %>%
    add_recipe(recipe) %>%
    add_model(lightgbm_spec)

  lightgbm_tune <-
    lightgbm_workflow %>%
    tune_grid(resamples = df_folds,
              grid = lightgbm_grid)

  lightgbm_tune %>%
    show_best("roc_auc") %>%
    niceFormatting(digits = 5, label = 4)

  best_lightgbm <- lightgbm_tune %>%
    select_best("roc_auc")

  autoplot(lightgbm_tune, metric = "roc_auc")
```

```
  final_lightgbm_workflow <-
    lightgbm_workflow %>%
    finalize_workflow(best_lightgbm)

  last_lightgbm_fit <-
    final_lightgbm_workflow %>%
    last_fit(df_split)

  final_lightgbm_fit <- extract_workflow(last_lightgbm_fit)

  lightgbm_auc <- validation(final_lightgbm_fit, df_test)

  lightgbm_parameters <- lightgbm_tune %>%
    show_best("roc_auc", n = 1) %>%
    select(trees, min_n, tree_depth, learn_rate) %>%
    as.list

  return(list(auc = as.numeric(lightgbm_auc$auc),
              auc_lower = lightgbm_auc$ci[1],
              auc_upper = lightgbm_auc$ci[3],
              parameters = lightgbm_parameters,
              fit = final_lightgbm_fit))
}

standard_results <- lightgbm_tuning(lightgbm_recipe)
```




```
## [1] "Optimal Threshold: 0.08"
## Confusion Matrix and Statistics
##
##     reference
## data    0    1
##    0 2995  182
##    1 1297  256
```

```
##
##                Accuracy : 0.6873
##                  95% CI : (0.6739, 0.7005)
##     No Information Rate : 0.9074
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1317
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.6978
##             Specificity : 0.5845
##          Pos Pred Value : 0.9427
##          Neg Pred Value : 0.1648
##              Prevalence : 0.9074
##          Detection Rate : 0.6332
##    Detection Prevalence : 0.6717
##       Balanced Accuracy : 0.6411
##
##        'Positive' Class : 0
##
```

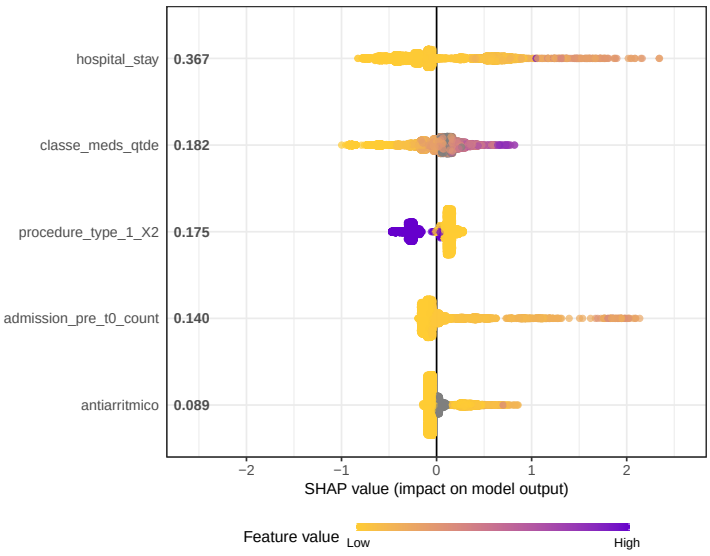
```
final_lightgbm_fit <- standard_results$fit
lightgbm_parameters <- standard_results$parameters

saveRDS(
  lightgbm_parameters,
  file = sprintf(
    "./auxiliar/final_model/hyperparameters/lightgbm_%s.rds",
    outcome_column
  )
)

# Save the final model. We need it for the calculator
lgb.save(
  parsnip::extract_fit_engine(final_lightgbm_fit),
  sprintf("./results/%s/final_model.txt", outcome_column)
)
saveRDS(final_lightgbm_fit,
        sprintf("./results/%s/final_model_wf.rds", outcome_column))
```

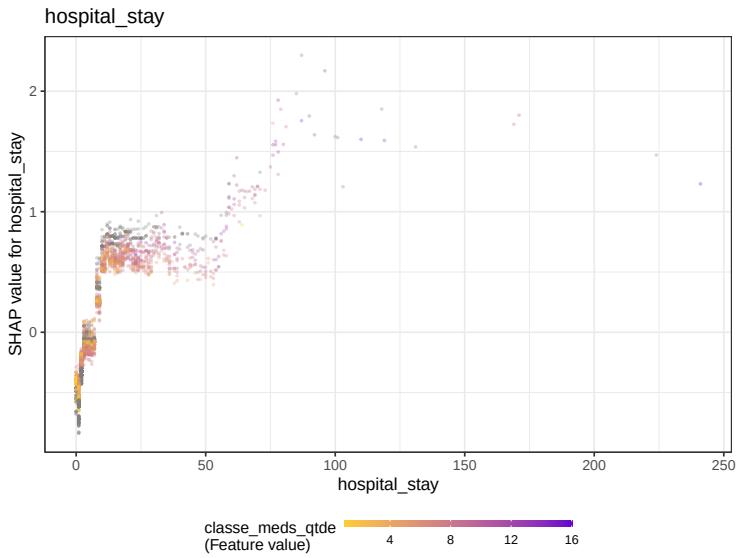
SHAP values

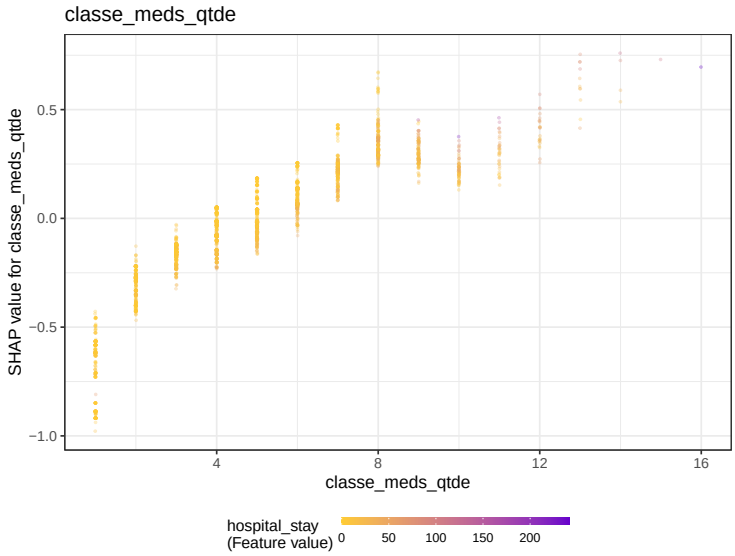
```
lightgbm_model <- parsnip::extract_fit_engine(final_lightgbm_fit)

trained_rec <- prep(lightgbm_recipe, training = df_train)

df_train_baked <- bake(trained_rec, new_data = df_train)
df_test_baked <- bake(trained_rec, new_data = df_test)

matrix_train <- as.matrix(df_train_baked %>% select(-all_of(outcome_column)))
matrix_test <- as.matrix(df_test_baked %>% select(-all_of(outcome_column)))

n_plots <- min(6, length(selected_features))
plotted <- 0

shap.plot.summary.wrap1(model = lightgbm_model, X = matrix_train,
                        top_n = n_plots, dilute = F)

shap <- shap.prep(lightgbm_model, X_train = matrix_test)
```

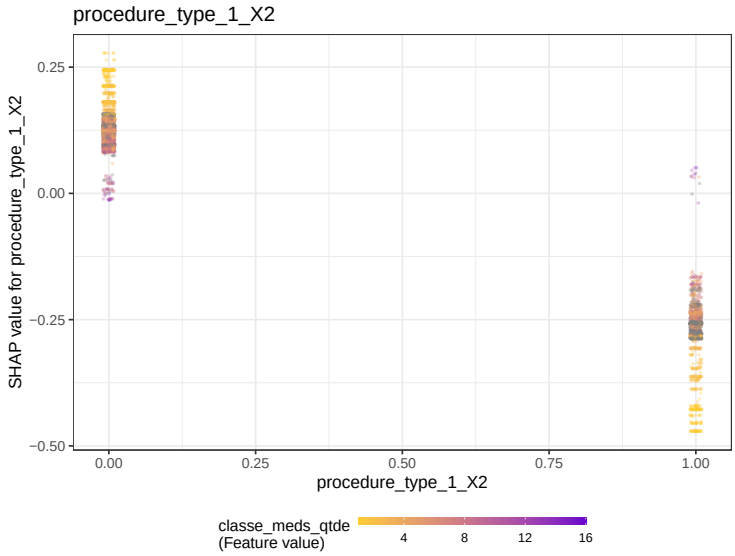
```r
for (x in shap.importance(shap, names_only = TRUE)) {
  p <- shap.plot.dependence(
    shap,
    x = x,
    color_feature = "auto",
    smooth = FALSE,
    jitter_width = 0.01,
    alpha = 0.3
  ) +
    labs(title = x)

  if (plotted < n_plots) {
    print(p)
    plotted <- plotted + 1
  }

  ggsave(sprintf("./auxiliar/final_model/shap_plots/%s.png", x),
         plot = p,
         dpi = 300)
}
```

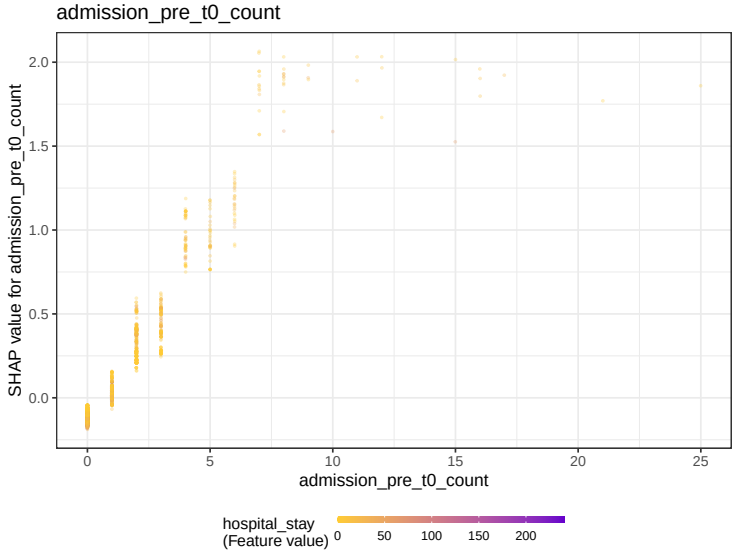
```
## Saving 6.5 x 5 in image
## Warning: Removed 1472 rows containing missing values (geom_point).
## Saving 6.5 x 5 in image
## Warning: Removed 1472 rows containing missing values (geom_point).
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Warning: Removed 1064 rows containing missing values (geom_point).
## Saving 6.5 x 5 in image
## Warning: Removed 1064 rows containing missing values (geom_point).
## Saving 6.5 x 5 in image
```

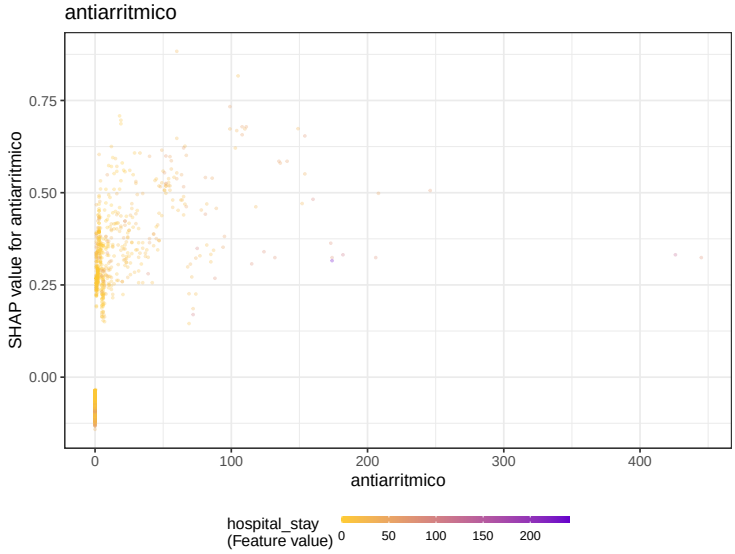










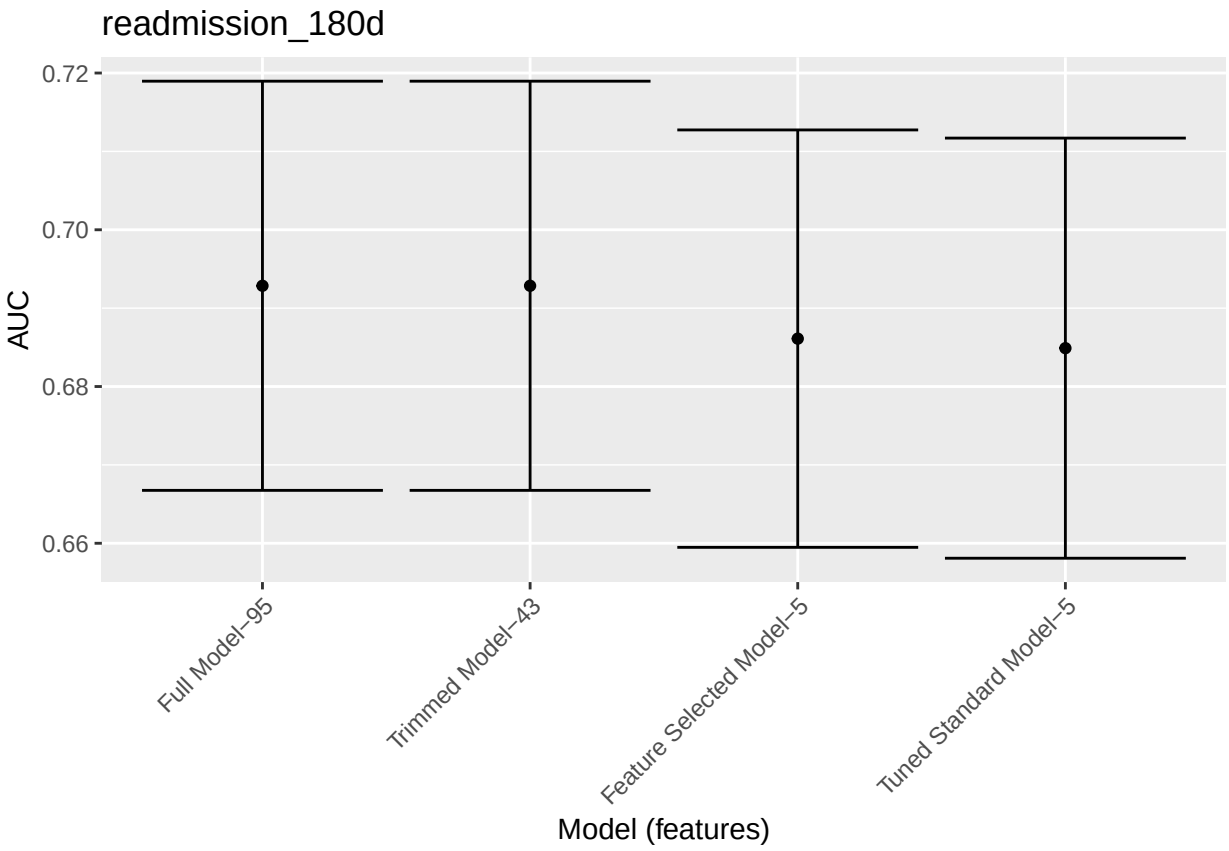



```
## $num_iterations
## [1] 108
##
## $learning_rate
## [1] 0.06726966
##
## $max_depth
## [1] 3
##
## $feature_fraction
## [1] 1
##
## $min_data_in_leaf
## [1] 78
##
## $min_gain_to_split
## [1] 0
##
## $bagging_fraction
## [1] 1
##
## $num_class
## [1] 1
##
## $objective
```

```
## [1] "binary"
##
## $num_threads
## $num_threads$num_threads
## [1] 0
##
##
## $nthread
## [1] 8
##
## $seed
## [1] 68582
##
## $deterministic
## [1] TRUE
##
## $verbose
## [1] -1
##
## $metric
## list()
##
## $interaction_constraints
## list()
##
## $feature_pre_filter
## [1] FALSE
```

Models Comparison

```
df_auc <- tibble::tribble(
  ~Model, ~`AUC`, ~`Lower Limit`, ~`Upper Limit`, ~`Features`,
  'Full Model', full_model$auc, full_model$auc_lower, full_model$auc_upper, length(features),
  'Trimmed Model', trimmed_model$auc, trimmed_model$auc_lower, trimmed_model$auc_upper, length(trimmed_features)
  'Feature Selected Model', feature_selected_model$auc, feature_selected_model$auc_lower, feature_selected_model
  'Tuned Standard Model', standard_results$auc, standard_results$auc_lower, standard_results$auc_upper, length(s
) %>%
  mutate(Target = outcome_column,
         `Model (features)` = fct_reorder(paste0(Model, "-", Features), -Features))

df_auc %>%
  ggplot(aes(
    x = `Model (features)`,
    y = AUC,
    ymin = `Lower Limit`,
    ymax = `Upper Limit`
  )) +
  geom_point() +
  geom_errorbar() +
  labs(title = outcome_column) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
saveRDS(df_auc, sprintf("./auxiliar/final_model/performance/%s.RData", outcome_column))
```