Final Model - death_3year

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 (952 bytes)
## .  max_auc_loss = double 1= 0.01
## .  outcome_column = character 1= death_3year
## .  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
predict <- stats::predict
```

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
```

```r
df[columns_list$outcome_columns] <- lapply(df[columns_list$outcome_columns], factor)
df <- mutate(df, across(where(is.character), as.factor))
```

```r
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)

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

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

Eligible features

```r
cat_features_list = read_yaml(sprintf(
  "./auxiliar/significant_columns/categorical_%s.yaml",
  outcome_column
))

num_features_list = read_yaml(sprintf(
  "./auxiliar/significant_columns/numerical_%s.yaml",
  outcome_column
))

features_list = c(cat_features_list, num_features_list)
```

```r
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. race
4. education_level
5. underlying_heart_disease
6. heart_disease
7. nyha_basal
8. hypertension
9. prior_mi
10. heart_failure
11. af
12. cardiac_arrest
13. valvopathy
14. diabetes
15. renal_failure
16. hemodialysis
17. stroke
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. year_adm_t0
28. icu_t0
29. dialysis_t0
30. admission_t0_emergency
31. aco
32. antiarritmico
33. ieca_bra
34. dva
35. digoxina
36. estatina
37. diuretico
38. vasodilatador
39. insuf_cardiaca
40. espironolactona
41. antiplaquetario_ev
42. insulina
43. anticonvulsivante
44. psicofarmacos
45. antifungico
46. classe_meds_qtde
47. meds_cardiovasc_qtde
48. meds_antimicrobianos
49. ventilacao_mecanica
50. transplante_cardiaco
51. outros_proced_cirurgicos
52. icp
53. angioplastia
54. cateterismo
55. eletrofisiologia
56. cateter_venoso_central
57. proced_invasivos_qtde
58. transfusao
59. equipe_multiprof

60. holter
61. teste_esforco
62. tilt_teste
63. metodos_graficos_qtde
64. laboratorio
65. cultura
66. analises_clinicas_qtde
67. citologia
68. histopatologia_qtde
69. angio_tc
70. angiografia
71. cintilografia
72. ecocardiograma
73. endoscopia
74. flebografia
75. pet_ct
76. ultrassom
77. tomografia
78. ressonancia
79. exames_imagem_qtde
80. bic
81. 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 <- read_yaml(
  sprintf(
    "./auxiliar/model_selection/hyperparameters/%s.yaml",
    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: 0.800"
```

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

```
## [1] "Full Model Test AUC: 0.809"
```

Features with zero importance on the initial model:

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

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

1. antiplaquetario_ev
2. transplante_cardiaco
3. angioplastia
4. eletrofisiologia
5. teste_esforco
6. tilt_teste
7. histopatologia_qtde
8. angiografia

```
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: 0.800"
```

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

```
## [1] "Trimmed Model Test AUC: 0.809"
```

```
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] "72"     "0.0017" "0.0015" "pet_ct"
## [1] "71"         "0.0019"     "1e-04"      "transfusao"
## [1] "70"       "0"        "-0.0019" "stroke"
## [1] "69"        "-1e-04"   "-1e-04"   "angio_tc"
## [1] "68"     "-1e-04" "0"      "holter"
## [1] "67"           "8e-04"        "9e-04"        "hemodialysis"
## [1] "66"             "7e-04"          "-1e-04"         "cardiac_arrest"
## [1] "65"          "9e-04"       "1e-04"       "flebografia"
## [1] "64"            "0"             "-8e-04"        "cintilografia"
## [1] "63"  "0"   "0"   "icp"
## [1] "62"          "0"           "0"           "reop_type_1"
## [1] "61"          "2e-04"       "2e-04"       "dialysis_t0"
## [1] "60"          "4e-04"       "2e-04"       "cateterismo"
## [1] "59"                      "-2e-04"                "-7e-04"
## [4] "cateter_venoso_central"
## [1] "58"                  "3e-04"              "5e-04"              "procedure_type_new"
## [1] "57"          "0.0012"      "9e-04"       "ressonancia"
## [1] "56"            "5e-04"         "-6e-04"        "heart_failure"
## [1] "55"                        "0.001"                   "4e-04"
## [4] "outros_proced_cirurgicos"
## [1] "54"     "6e-04"  "-4e-04" "copd"
## [1] "53"     "-2e-04" "-8e-04" "aco"
## [1] "52"       "0.0016"   "0.0018"   "insulina"
## [1] "51"          "0.0016"      "0"           "antifungico"
## [1] "50"        "0.001"     "-6e-04"    "citologia"
## [1] "49"         "0.0011"     "1e-04"      "endoscopia"
## [1] "48"       "6e-04"    "-4e-04"   "prior_mi"
## [1] "47"                     "0.0013"                "7e-04"
## [4] "analises_clinicas_qtde"
```

```
## [1] "46"            "4e-04"         "-9e-04"        "heart_disease"
## [1] "46"                 "4e-04"              "0.0025"             "ventilacao_mecanica"
## [1] "45"                   "0.001"                "6e-04"
## [4] "admission_pre_t0_180d"
## [1] "44"        "-3e-04"    "-0.0013"   "ultrassom"
## [1] "43"     "0.001"  "0.0013" "bic"
## [1] "42"             "0.0012"         "2e-04"          "ecocardiograma"
## [1] "41"           "-0.0011"      "-0.0022"      "hypertension"
## [1] "40"     "7e-04"  "0.0017" "sex"
## [1] "39"                     "-7e-04"                "-0.0013"
## [4] "admission_t0_emergency"
## [1] "38"                "3e-04"             "0.001"             "anticonvulsivante"
## [1] "37"               "-0.001"           "-0.0013"          "procedure_type_1"
## [1] "36"     "1e-04"  "0.0011" "dva"
## [1] "35"       "-7e-04"   "-8e-04"   "diabetes"
## [1] "34"                   "-2e-04"               "5e-04"
## [4] "proced_invasivos_qtde"
## [1] "33"          "-0.0017"    "-0.0015"    "tomografia"
## [1] "32"      "-0.0013" "4e-04"   "cultura"
## [1] "31"       "-0.0023"  "-0.001"   "digoxina"
## [1] "30"     "-5e-04" "0.0018" "af"
## [1] "29"         "-0.001"     "-5e-04"     "valvopathy"
## [1] "28"           "-9e-04"       "1e-04"        "cied_final_1"
## [1] "27"            "-0.0018"       "-9e-04"        "renal_failure"
## [1] "26"     "-7e-04" "0.0011" "race"
## [1] "26"                "-7e-04"             "0.0033"             "cied_final_group_1"
## [1] "26"            "-7e-04"        "0.0021"        "antiarritmico"
## [1] "25"       "1e-04"    "8e-04"    "estatina"
## [1] "24"                       "0.0014"                   "0.0013"
## [4] "underlying_heart_disease"
## [1] "23"                "7e-04"              "-7e-04"             "exames_imagem_qtde"
## [1] "22"     "7e-04"  "0"      "icu_t0"
## [1] "21"               "6e-04"            "0"                "equipe_multiprof"
## [1] "20"               "7e-04"            "0"                "classe_meds_qtde"
## [1] "19"                   "1e-04"                "-5e-04"               "meds_antimicrobianos"
## [1] "18"            "9e-04"         "8e-04"         "vasodilatador"
## [1] "17"             "-1e-04"         "-0.001"         "insuf_cardiaca"
## [1] "16"            "9e-04"         "9e-04"         "psicofarmacos"
## [1] "16"         "9e-04"      "0.0055"     "nyha_basal"
## [1] "15"                    "0.0025"               "0.0016"
## [4] "metodos_graficos_qtde"
## [1] "14"        "0.0025"    "0"         "diuretico"
## [1] "13"          "-0.0035"     "-0.006"      "laboratorio"
## [1] "13"                  "-0.0035"            "0.0057"             "comorbidities_count"
## [1] "13"       "-0.0035"  "0.0035"   "ieca_bra"
## [1] "13"               "-0.0035"         "0.0107"          "education_level"
## [1] "13"                    "-0.0035"              "0.0022"               "meds_cardiovasc_qtde"
## [1] "13"               "-0.0035"         "0.0116"          "espironolactona"
## [1] "13"                      "-0.0035"               "0.0174"
## [4] "admission_pre_t0_count"
## [1] "13"          "-0.0035"     "0.0222"      "year_adm_t0"
## [1] "13"      "-0.0035" "0.0082"  "age"
## [1] "13"            "-0.0035"       "0.0236"        "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	81	0.7998	0.0057	0.0000	0.0000
All unimportant	TRUE	73	0.7996	0.0058	0.0002	0.0002
pet_ct	TRUE	72	0.7981	0.0058	0.0017	0.0015
transfusao	TRUE	71	0.7979	0.0057	0.0019	0.0001
stroke	TRUE	70	0.7998	0.0058	0.0000	-0.0019
angio_tc	TRUE	69	0.7999	0.0059	-0.0001	-0.0001
holter	TRUE	68	0.7999	0.0057	-0.0001	0.0000
hemodialysis	TRUE	67	0.7990	0.0059	0.0008	0.0009
cardiac_arrest	TRUE	66	0.7991	0.0059	0.0007	-0.0001
flebografia	TRUE	65	0.7990	0.0058	0.0009	0.0001
cintilografia	TRUE	64	0.7998	0.0056	0.0000	-0.0008
icp	TRUE	63	0.7998	0.0056	0.0000	0.0000
reop_type_1	TRUE	62	0.7998	0.0056	0.0000	0.0000
dialysis_t0	TRUE	61	0.7996	0.0058	0.0002	0.0002
cateterismo	TRUE	60	0.7994	0.0056	0.0004	0.0002
cateter_venoso_central	TRUE	59	0.8000	0.0058	-0.0002	-0.0007
procedure_type_new	TRUE	58	0.7995	0.0061	0.0003	0.0005
ressonancia	TRUE	57	0.7986	0.0054	0.0012	0.0009
heart_failure	TRUE	56	0.7993	0.0057	0.0005	-0.0006
outros_proced_cirurgicos	TRUE	55	0.7988	0.0057	0.0010	0.0004
copd	TRUE	54	0.7992	0.0059	0.0006	-0.0004
aco	TRUE	53	0.8000	0.0061	-0.0002	-0.0008
insulina	TRUE	52	0.7982	0.0060	0.0016	0.0018
antifungico	TRUE	51	0.7982	0.0058	0.0016	0.0000
citologia	TRUE	50	0.7988	0.0060	0.0010	-0.0006
endoscopia	TRUE	49	0.7987	0.0057	0.0011	0.0001
prior_mi	TRUE	48	0.7992	0.0060	0.0006	-0.0004
analises_clinicas_qtde	TRUE	47	0.7985	0.0060	0.0013	0.0007
heart_disease	TRUE	46	0.7994	0.0060	0.0004	-0.0009
ventilacao_mecanica	FALSE	45	0.7969	0.0058	0.0004	0.0025
admission_pre_t0_180d	TRUE	45	0.7988	0.0059	0.0010	0.0006
ultrassom	TRUE	44	0.8001	0.0060	-0.0003	-0.0013
bic	TRUE	43	0.7988	0.0059	0.0010	0.0013
ecocardiograma	TRUE	42	0.7987	0.0058	0.0012	0.0002
hypertension	TRUE	41	0.8009	0.0055	-0.0011	-0.0022
sex	TRUE	40	0.7991	0.0059	0.0007	0.0017
admission_t0_emergency	TRUE	39	0.8005	0.0057	-0.0007	-0.0013
anticonvulsivante	TRUE	38	0.7995	0.0059	0.0003	0.0010
procedure_type_1	TRUE	37	0.8008	0.0059	-0.0010	-0.0013
dva	TRUE	36	0.7998	0.0060	0.0001	0.0011
diabetes	TRUE	35	0.8005	0.0061	-0.0007	-0.0008
proced_invasivos_qtde	TRUE	34	0.8000	0.0056	-0.0002	0.0005
tomografia	TRUE	33	0.8015	0.0058	-0.0017	-0.0015
cultura	TRUE	32	0.8011	0.0062	-0.0013	0.0004
digoxina	TRUE	31	0.8021	0.0056	-0.0023	-0.0010
af	TRUE	30	0.8003	0.0055	-0.0005	0.0018
valvopathy	TRUE	29	0.8008	0.0057	-0.0010	-0.0005
cied_final_1	TRUE	28	0.8007	0.0058	-0.0009	0.0001
renal_failure	TRUE	27	0.8016	0.0061	-0.0018	-0.0009
race	TRUE	26	0.8005	0.0058	-0.0007	0.0011
cied_final_group_1	FALSE	25	0.7973	0.0058	-0.0007	0.0033
antiarritmico	FALSE	25	0.7984	0.0062	-0.0007	0.0021
estatina	TRUE	25	0.7997	0.0056	0.0001	0.0008

Table 1: *(continued)*

Tested Feature	Dropped	Features	CV AUC	CV AUC Std Error	Total AUC Loss	Instant AUC Loss
underlying_heart_disease	TRUE	24	0.7984	0.0057	0.0014	0.0013
exames_imagem_qtde	TRUE	23	0.7991	0.0057	0.0007	-0.0007
icu_t0	TRUE	22	0.7992	0.0058	0.0007	0.0000
equipe_multiprof	TRUE	21	0.7992	0.0055	0.0006	0.0000
classe_meds_qtde	TRUE	20	0.7992	0.0057	0.0007	0.0000
meds_antimicrobianos	TRUE	19	0.7997	0.0057	0.0001	-0.0005
vasodilatador	TRUE	18	0.7989	0.0056	0.0009	0.0008
insuf_cardiaca	TRUE	17	0.7999	0.0055	-0.0001	-0.0010
psicofarmacos	TRUE	16	0.7990	0.0056	0.0009	0.0009
nyha_basal	FALSE	15	0.7935	0.0060	0.0009	0.0055
metodos_graficos_qtde	TRUE	15	0.7973	0.0057	0.0025	0.0016
diuretico	TRUE	14	0.7973	0.0057	0.0025	0.0000
laboratorio	TRUE	13	0.8034	0.0057	-0.0035	-0.0060
comorbidities_count	FALSE	12	0.7976	0.0061	-0.0035	0.0057
ieca_bra	FALSE	12	0.7999	0.0053	-0.0035	0.0035
education_level	FALSE	12	0.7927	0.0058	-0.0035	0.0107
meds_cardiovasc_qtde	FALSE	12	0.8011	0.0060	-0.0035	0.0022
espironolactona	FALSE	12	0.7918	0.0054	-0.0035	0.0116
admission_pre_t0_count	FALSE	12	0.7860	0.0057	-0.0035	0.0174
year_adm_t0	FALSE	12	0.7812	0.0056	-0.0035	0.0222
age	FALSE	12	0.7951	0.0056	-0.0035	0.0082
hospital_stay	FALSE	12	0.7798	0.0061	-0.0035	0.0236

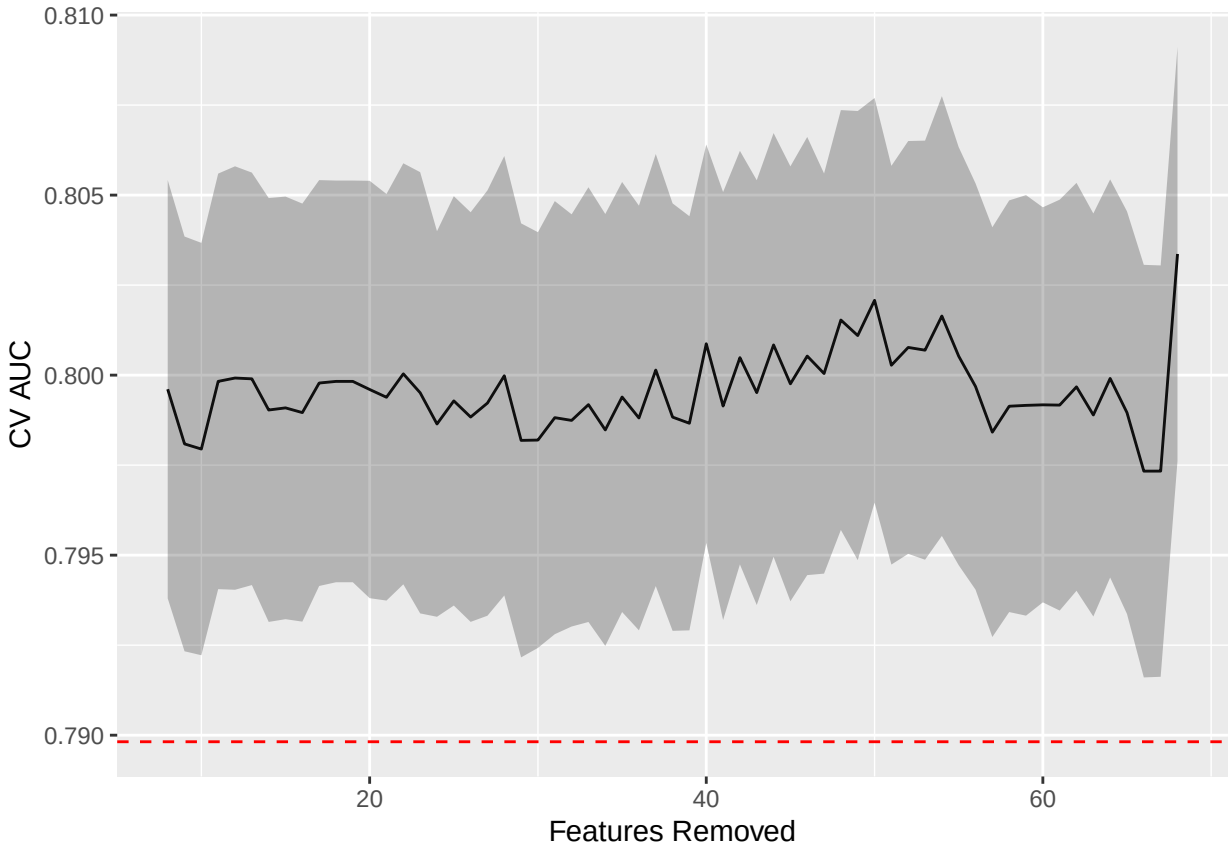
```
selected_features <- current_features

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: 0.803"
```

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

```
## [1] "Selected Model Test AUC: 0.797"
```

```
 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)
```

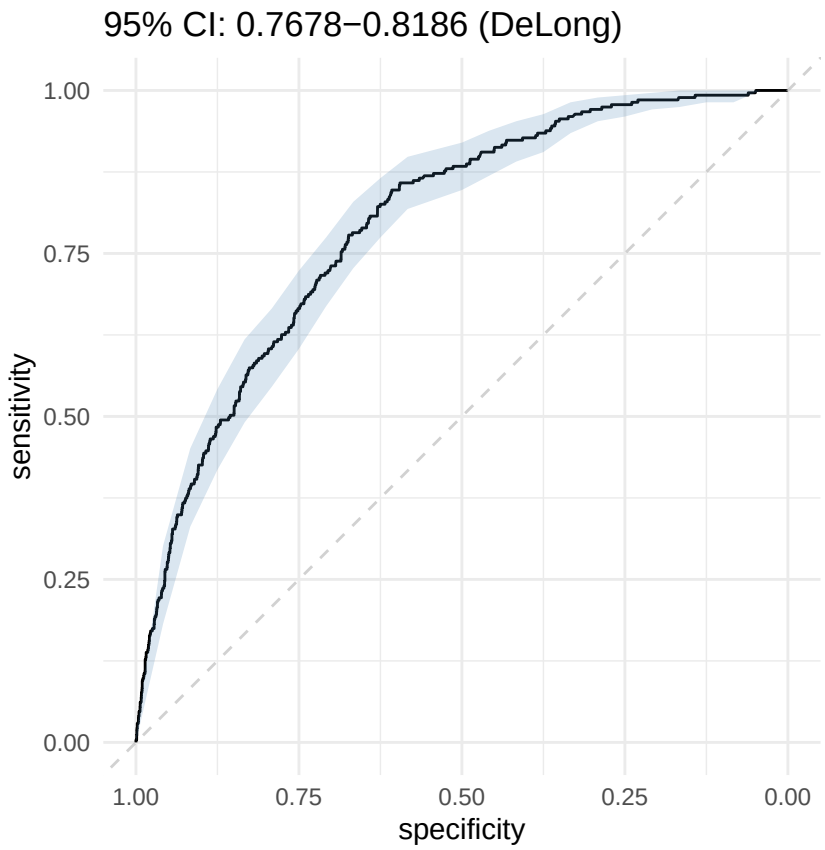
1. hospital_stay
2. age
3. year_adm_t0
4. admission_pre_t0_count
5. espironolactona
6. education_level
7. comorbidities_count
8. ieca_bra
9. meds_cardiovasc_qtde
10. nyha_basal
11. antiarritmico
12. cied_final_group_1
13. ventilacao_mecanica

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: 0.04"
## Confusion Matrix and Statistics
##
##     reference
## data    0    1
##    0 2707   42
##    1 1748  233
##
##                Accuracy : 0.6216
##                  95% CI : (0.6076, 0.6354)
##     No Information Rate : 0.9419
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1163
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.6076
##             Specificity : 0.8473
##          Pos Pred Value : 0.9847
##          Neg Pred Value : 0.1176
##              Prevalence : 0.9419
##          Detection Rate : 0.5723
##    Detection Prevalence : 0.5812
##       Balanced Accuracy : 0.7275
##
##        'Positive' Class : 0
##
```

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

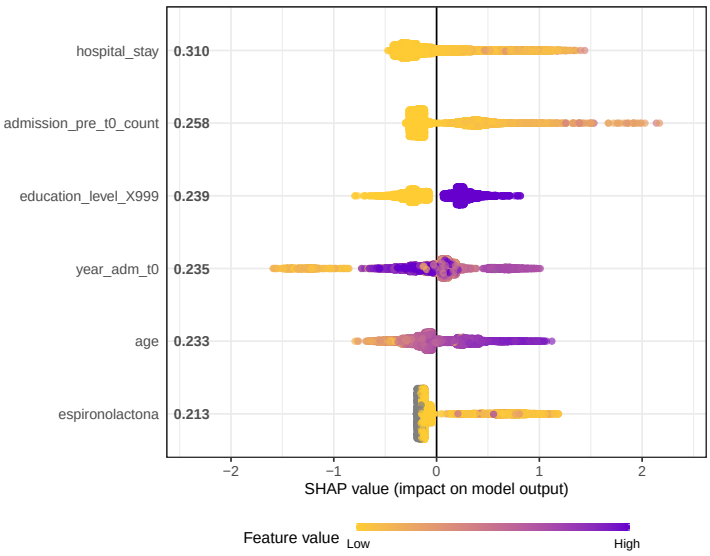
```
con <- file(sprintf('./auxiliar/final_model/hyperparameters/%s.yaml',
                    outcome_column), "w")
write_yaml(lightgbm_parameters, con)
close(con)

# 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))
```

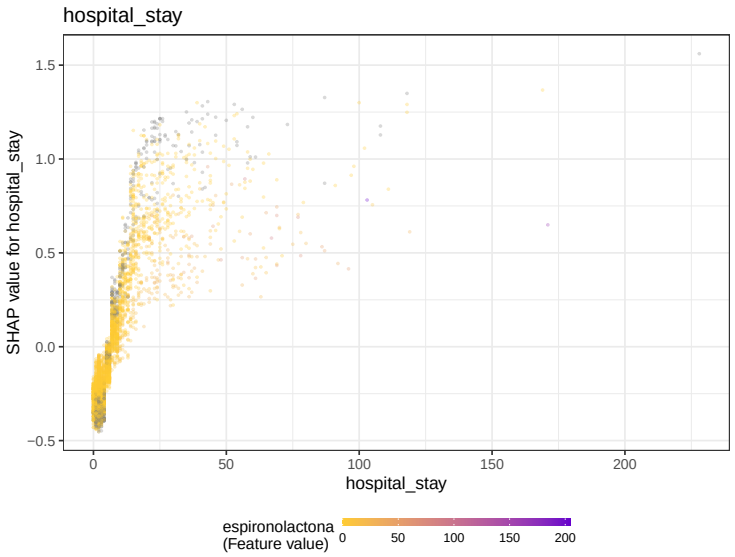
SHAP values

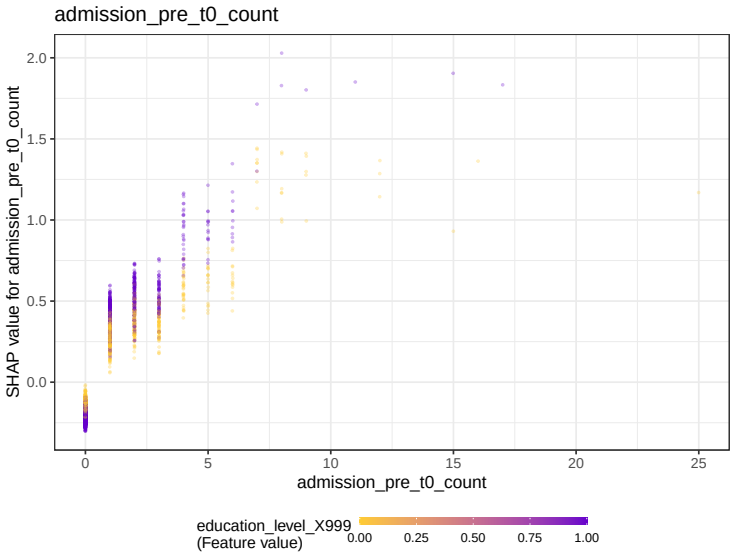
```
lightgbm_model <- parsnip::extract_fit_engine(final_lightgbm_fit)

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)

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/%s.png",
                 outcome_column, x),
         plot = p,
         dpi = 300)
}
```

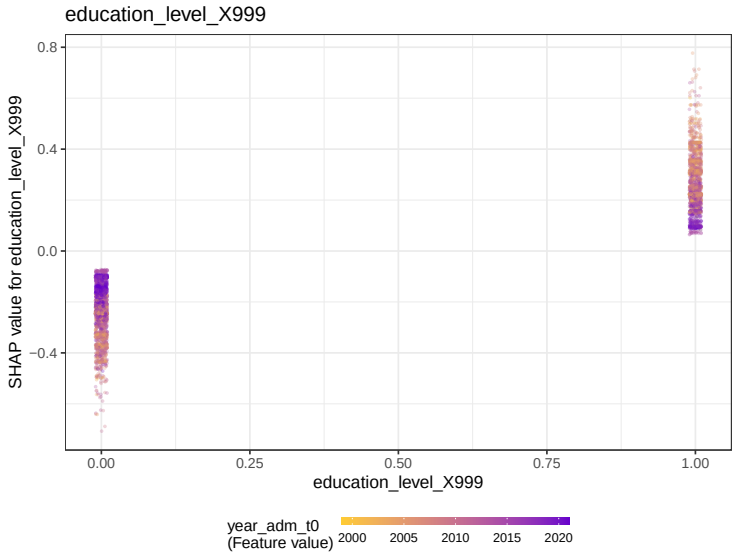
```
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
```

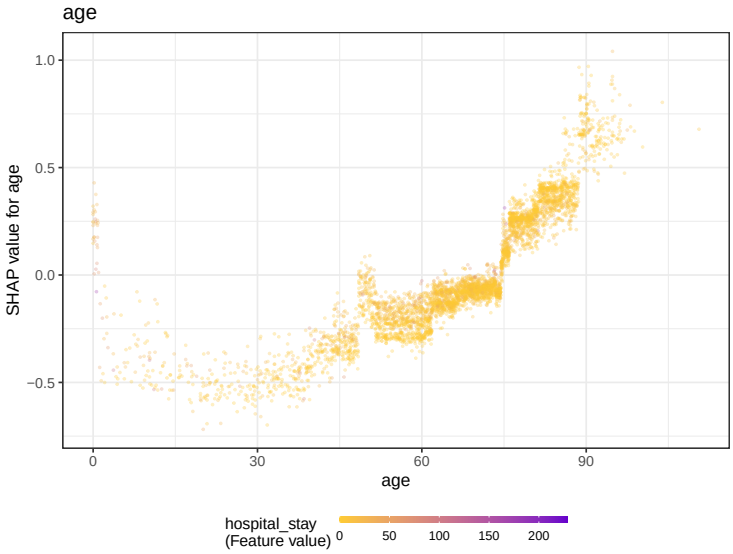
```
## Warning: Removed 6 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Warning: Removed 6 rows containing missing values (‘geom_point()‘).
## Warning: Removed 1070 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Warning: Removed 1070 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Warning: Removed 1070 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Warning: Removed 1070 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Warning: Removed 1070 rows containing missing values (‘geom_point()‘).
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
## Saving 6.5 x 5 in image
```

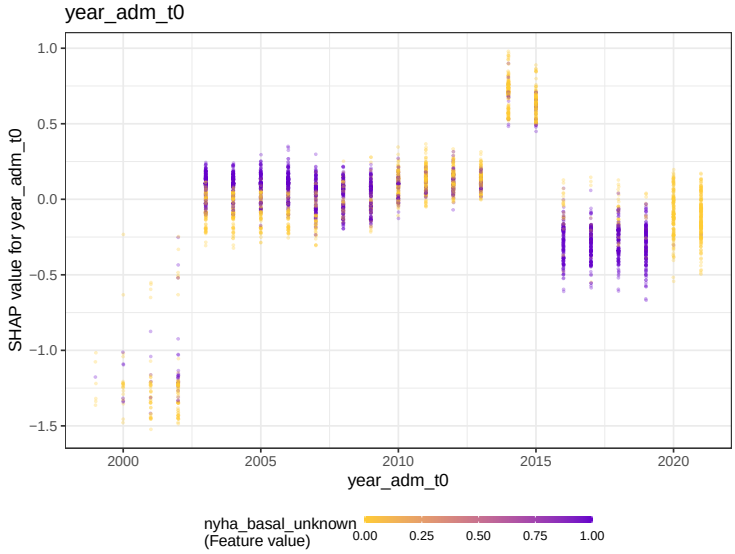


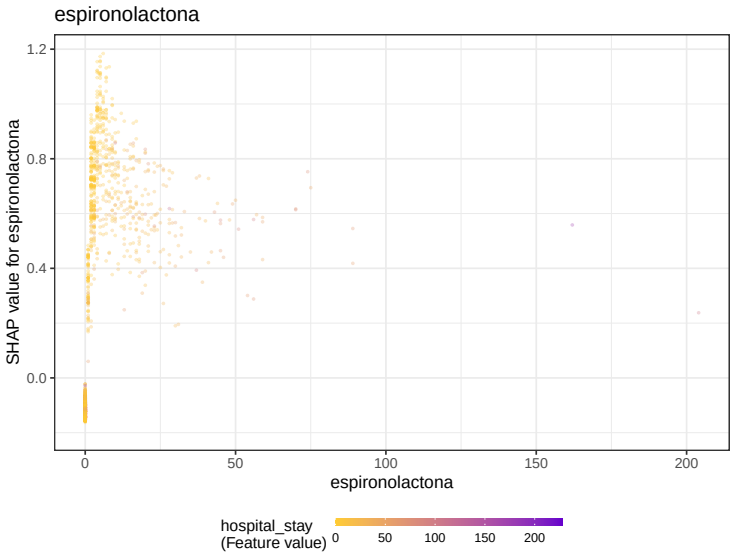












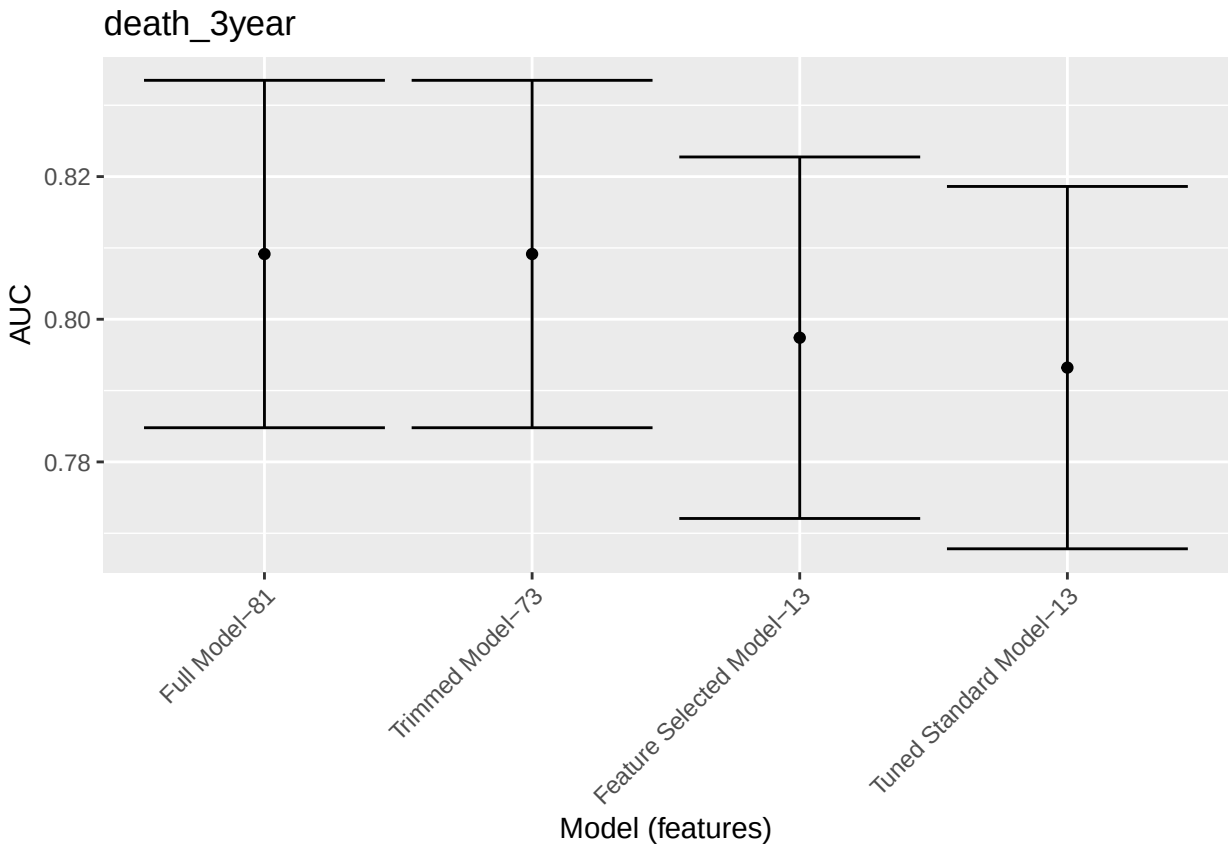

```
## $num_iterations
## [1] 93
##
## $learning_rate
## [1] 0.07108059
##
## $max_depth
```

```
## [1] 3
##
## $feature_fraction_bynode
## [1] 1
##
## $min_data_in_leaf
## [1] 34
##
## $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] 23005
##
## $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))
```




```
write_csv(df_auc, sprintf("./auxiliar/final_model/performance/%s.csv", outcome_column))
```