

Final Model - death_2year

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_2year
## . 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
```

```

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 = 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)

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. underlying_heart_disease
5. heart_disease
6. nyha_basal

7. hypertension
8. prior_mi
9. heart_failure
10. af
11. cardiac_arrest
12. valvopathy
13. diabetes
14. renal_failure
15. hemodialysis
16. stroke
17. copd
18. comorbidities_count
19. procedure_type_1
20. reop_type_1
21. procedure_type_new
22. cied_final_1
23. cied_final_group_1
24. admission_pre_t0_count
25. admission_pre_t0_180d
26. year_adm_t0
27. icu_t0
28. dialysis_t0
29. admission_t0_emergency
30. aco
31. antiarritmico
32. ieca_bra
33. dva
34. digoxina
35. estatina
36. diuretico
37. vasodilatador
38. insuf_cardiaca
39. espironolactona
40. antiplaquetario_ev
41. insulina
42. psicofarmacos
43. antifungico
44. antiviral
45. classe_meds_qtde
46. meds_cardiovasc_qtde
47. meds_antimicrobianos
48. vni
49. ventilacao_mecanica
50. transplante_cardiaco
51. outros_proced_cirurgicos
52. icp
53. angioplastia
54. cateterismo
55. cateter Venoso Central
56. proced_invasivos_qtde
57. transfusao
58. interconsulta
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. cintilografia
 71. ecocardiograma
 72. endoscopia
 73. flebografia
 74. pet_ct
 75. ultrassom
 76. tomografia
 77. ressonancia
 78. exames_imagem_qtde
 79. bic
 80. 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: 0.803"
sprintf('Full Model Test AUC: %.3f' ,full_model$auc)

## [1] "Full Model Test AUC: 0.802"

Features with zero importance on the initial model:

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

unimportant_features %>%
  gluedown::md_order()

1. heart_failure
2. hemodialysis
3. stroke
4. reop_type_1
5. dialysis_t0
6. antiplaquetario_ev
7. antifungico

```

```

8. antiviral
9. vni
10. transplante_cardiaco
11. outros_proced_cirurgicos
12. icp
13. angioplastia
14. cateter_venoso_central
15. transfusao
16. teste_esforco
17. tilt_teste
18. histopatologia_qtde
19. angio_tc
20. endoscopia
21. flebografia
22. pet_ct

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: 0.803"
sprintf('Trimmed Model Test AUC: %.3f' ,trimmed_model$auc)

## [1] "Trimmed Model Test AUC: 0.802"
selection_results <- tibble::tribble(
  ~`Tested Feature`, ~`Dropped`, ~`Number of Features`, ~`CV AUC`, ~`CV AUC Std Error`, ~`Total AUC Loss`, ~`Instant AUC Loss`,
  '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] "57"                      "0"                      "5e-04"
## [4] "admission_t0_emergency"
## [1] "56"      "-5e-04"    "-5e-04"    "digoxina"
## [1] "55"      "-5e-04"    "0"        "insulina"
## [1] "54"      "-5e-04"    "0"        "holter"
## [1] "53"      "-3e-04"    "2e-04"    "copd"
## [1] "52"      "-4e-04"    "-1e-04"   "tomografia"
## [1] "51"      "-6e-04"    "-2e-04"   "cardiac_arrest"
## [1] "50"      "-5e-04"    "2e-04"    "cateterismo"
## [1] "49"      "-0.001"   "-5e-04"   "citologia"
## [1] "48"      "-0.001"   "-1e-04"   "bic"
## [1] "47"      "-8e-04"   "3e-04"    "procedure_type_1"
## [1] "46"      "-0.001"   "-3e-04"   "ressonancia"
## [1] "45"      "-5e-04"   "6e-04"    "ecocardiograma"
## [1] "44"      "-0.0012"  "-8e-04"   "prior_mi"
## [1] "43"      "-0.0016"  "-4e-04"   "procedure_type_new"
## [1] "42"      "-0.0019"  "-3e-04"   "sex"
## [1] "41"      "-0.002"   "-1e-04"   "heart_disease"
## [1] "40"      "-0.0013"  "7e-04"    "ultrassom"
## [1] "39"      "-8e-04"   "5e-04"    "ventilacao_mecanica"
## [1] "38"      "-0.001"   "-2e-04"   "hypertension"
## [1] "37"      "-0.0011"  "-1e-04"   "cultura"
## [1] "36"      "-0.0019"  "-7e-04"   "cied_final_1"

```

```

## [1] "35"           "-0.0015"          "4e-04"
## [4] "admission_pre_t0_180d"
## [1] "34"           "-0.002"   "-6e-04" "aco"
## [1] "33"           "-0.0016"  "4e-04"   "diabetes"
## [1] "32"           "-0.0016"  "0"        "interconsulta"
## [1] "31"           "-0.0019"  "-3e-04"  "renal_failure"
## [1] "30"           "-0.0019"  "0"        "af"
## [1] "29"           "-0.0021"  "-2e-04"  "cintilografia"
## [1] "28"           "-0.0022"  "-1e-04"
## [4] "analises_clinicas_qtde"
## [1] "27"           "-0.002"   "2e-04"   "valvopathy"
## [1] "26"           "-0.002"   "0"
## [4] "proced_invasivos_qtde"
## [1] "25"           "-0.0011"  "9e-04"   "classe_meds_qtde"
## [1] "24"           "-0.0012"  "-1e-04"  "exames_imagem_qtde"
## [1] "23"           "-0.0013"  "-1e-04"  "equipe_multiprof"
## [1] "22"           "-8e-04"   "6e-04"   "dva"
## [1] "21"           "-0.0027"  "-0.0019"  "insuf_cardiaca"
## [1] "20"           "-0.0024"  "2e-04"   "psicofarmacos"
## [1] "19"           "-0.0033"  "-9e-04"   "meds_cardiovasc_qtde"
## [1] "18"           "-0.0023"  "0.001"
## [4] "underlying_heart_disease"
## [1] "17"           "-0.0035"  "-0.0012"  "ieca_bra"
## [1] "17"           "-0.0035"  "0.0037"   "antiarritmico"
## [1] "16"           "-0.0032"  "3e-04"   "estatina"
## [1] "15"           "-0.0028"  "4e-04"   "meds_antimicrobianos"
## [1] "15"           "-0.0028"  "0.0031"   "cied_final_group_1"
## [1] "15"           "-0.0028"  "0.0027"
## [4] "metodos_graficos_qtde"
## [1] "14"           "-0.002"   "8e-04"   "icu_t0"
## [1] "14"           "-0.002"   "0.0041"   "diuretico"
## [1] "13"           "-0.0012"  "8e-04"   "vasodilatador"
## [1] "12"           "8e-04"   "0.002"   "nyha_basal"
## [1] "12"           "8e-04"   "0.0042"   "comorbidities_count"
## [1] "12"           "8e-04"   "0.0124"   "education_level"
## [1] "11"           "0.0027"  "0.002"   "laboratorio"
## [1] "11"           "0.0027"  "0.0043"   "espironolactona"
## [1] "11"           "0.0027"  "0.0155"
## [4] "admission_pre_t0_count"
## [1] "11"           "0.0027"  "0.0183"   "year_adm_t0"
## [1] "11"           "0.0027"  "0.0113"   "age"
## [1] "11"           "0.0027"  "0.0104"   "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	80	0.8028	0.0067	0.0000	0.0000
All unimportant	TRUE	58	0.8033	0.0067	-0.0005	-0.0005
admission_t0_emergency	TRUE	57	0.8028	0.0068	0.0000	0.0005
digoxina	TRUE	56	0.8033	0.0067	-0.0005	-0.0005
insulina	TRUE	55	0.8033	0.0068	-0.0005	0.0000
holter	TRUE	54	0.8033	0.0067	-0.0005	0.0000
copd	TRUE	53	0.8031	0.0067	-0.0003	0.0002
tomografia	TRUE	52	0.8032	0.0068	-0.0004	-0.0001
cardiac_arrest	TRUE	51	0.8034	0.0067	-0.0006	-0.0002
cateterismo	TRUE	50	0.8033	0.0066	-0.0005	0.0002

Table 1: (continued)

Tested Feature	Dropped	Features	CV AUC	CV AUC Std Error	Total AUC Loss	Instant AUC Loss
citologia	TRUE	49	0.8038	0.0066	-0.0010	-0.0005
bic	TRUE	48	0.8038	0.0066	-0.0010	-0.0001
procedure_type_1	TRUE	47	0.8036	0.0067	-0.0008	0.0003
ressonancia	TRUE	46	0.8039	0.0067	-0.0010	-0.0003
ecocardiograma	TRUE	45	0.8033	0.0067	-0.0005	0.0006
prior_mi	TRUE	44	0.8040	0.0068	-0.0012	-0.0008
procedure_type_new	TRUE	43	0.8044	0.0067	-0.0016	-0.0004
sex	TRUE	42	0.8047	0.0067	-0.0019	-0.0003
heart_disease	TRUE	41	0.8048	0.0068	-0.0020	-0.0001
ultrassom	TRUE	40	0.8041	0.0067	-0.0013	0.0007
ventilacao_mecanica	TRUE	39	0.8036	0.0068	-0.0008	0.0005
hypertension	TRUE	38	0.8038	0.0068	-0.0010	-0.0002
cultura	TRUE	37	0.8039	0.0067	-0.0011	-0.0001
cied_final_1	TRUE	36	0.8047	0.0067	-0.0019	-0.0007
admission_pre_t0_180d	TRUE	35	0.8043	0.0067	-0.0015	0.0004
aco	TRUE	34	0.8048	0.0066	-0.0020	-0.0006
diabetes	TRUE	33	0.8044	0.0068	-0.0016	0.0004
interconsulta	TRUE	32	0.8044	0.0067	-0.0016	0.0000
renal_failure	TRUE	31	0.8047	0.0068	-0.0019	-0.0003
af	TRUE	30	0.8047	0.0067	-0.0019	0.0000
cintilografia	TRUE	29	0.8049	0.0064	-0.0021	-0.0002
analises_clinicas_qtde	TRUE	28	0.8050	0.0067	-0.0022	-0.0001
valvopathy	TRUE	27	0.8048	0.0067	-0.0020	0.0002
proced_invasivos_qtde	TRUE	26	0.8048	0.0068	-0.0020	0.0000
classe_meds_qtde	TRUE	25	0.8039	0.0068	-0.0011	0.0009
exames_imagem_qtde	TRUE	24	0.8040	0.0070	-0.0012	-0.0001
equipe_multiprof	TRUE	23	0.8041	0.0068	-0.0013	-0.0001
dva	TRUE	22	0.8036	0.0071	-0.0008	0.0006
insuf_cardiaca	TRUE	21	0.8055	0.0069	-0.0027	-0.0019
psicofarmacos	TRUE	20	0.8053	0.0067	-0.0024	0.0002
meds_cardiovasc_qtde	TRUE	19	0.8062	0.0069	-0.0033	-0.0009
underlying_heart_disease	TRUE	18	0.8051	0.0068	-0.0023	0.0010
ieca_bra	TRUE	17	0.8063	0.0066	-0.0035	-0.0012
antiarritmico	FALSE	16	0.8026	0.0068	-0.0035	0.0037
estatina	TRUE	16	0.8060	0.0067	-0.0032	0.0003
meds_antimicrobianos	TRUE	15	0.8056	0.0067	-0.0028	0.0004
cied_final_group_1	FALSE	14	0.8026	0.0070	-0.0028	0.0031
metodos_graficos_qtde	FALSE	14	0.8029	0.0066	-0.0028	0.0027
icu_t0	TRUE	14	0.8048	0.0070	-0.0020	0.0008
diuretico	FALSE	13	0.8007	0.0069	-0.0020	0.0041
vasodilatador	TRUE	13	0.8040	0.0069	-0.0012	0.0008
nyha_basal	TRUE	12	0.8021	0.0063	0.0008	0.0020
comorbidities_count	FALSE	11	0.7978	0.0076	0.0008	0.0042
education_level	FALSE	11	0.7896	0.0067	0.0008	0.0124
laboratorio	TRUE	11	0.8001	0.0062	0.0027	0.0020
espironolactona	FALSE	10	0.7958	0.0055	0.0027	0.0043
admission_pre_t0_count	FALSE	10	0.7846	0.0061	0.0027	0.0155
year_adm_t0	FALSE	10	0.7818	0.0065	0.0027	0.0183
age	FALSE	10	0.7888	0.0063	0.0027	0.0113
hospital_stay	FALSE	10	0.7897	0.0077	0.0027	0.0104

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)

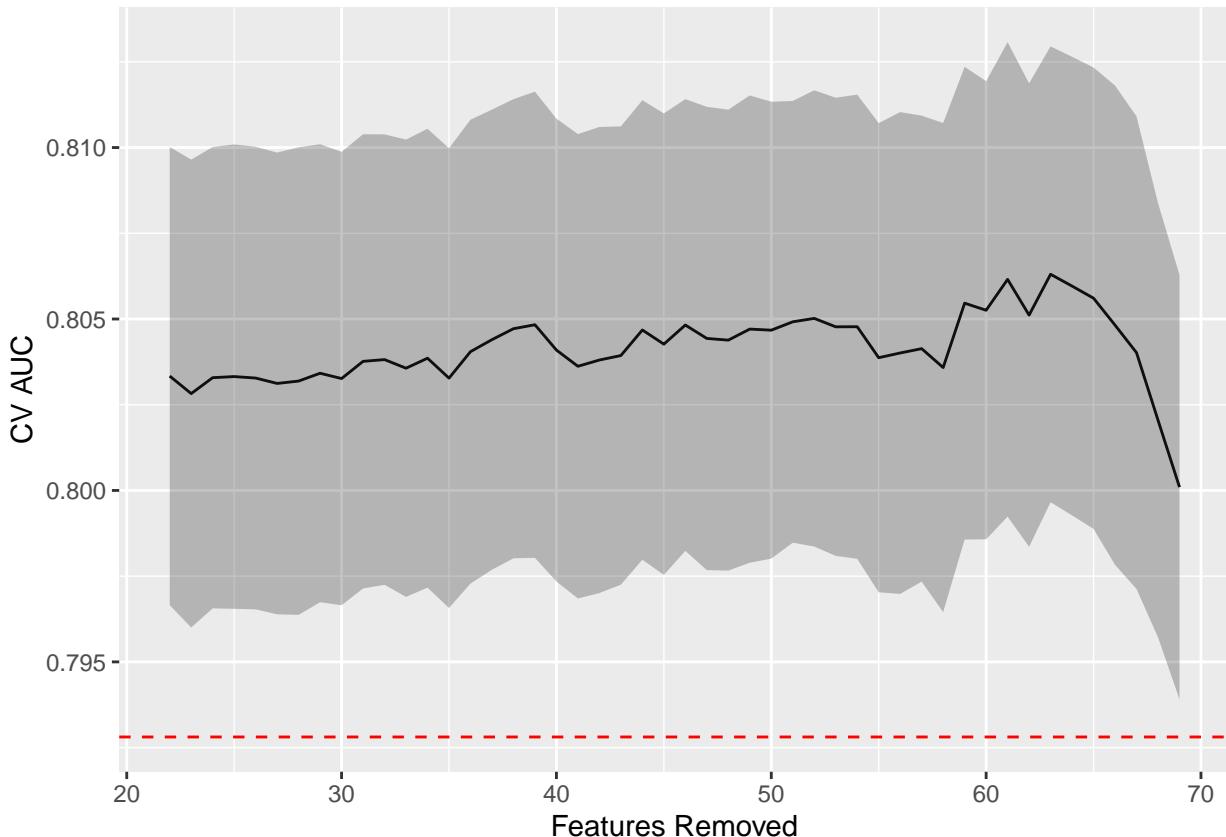
## [1] "Selected Model CV Train AUC: 0.800"
sprintf('Selected Model Test AUC: %.3f', feature_selected_model$auc)

## [1] "Selected Model Test AUC: 0.793"

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. diuretico
9. metodos_graficos_qtdc
10. cied_final_group_1
11. antiaritmico

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)

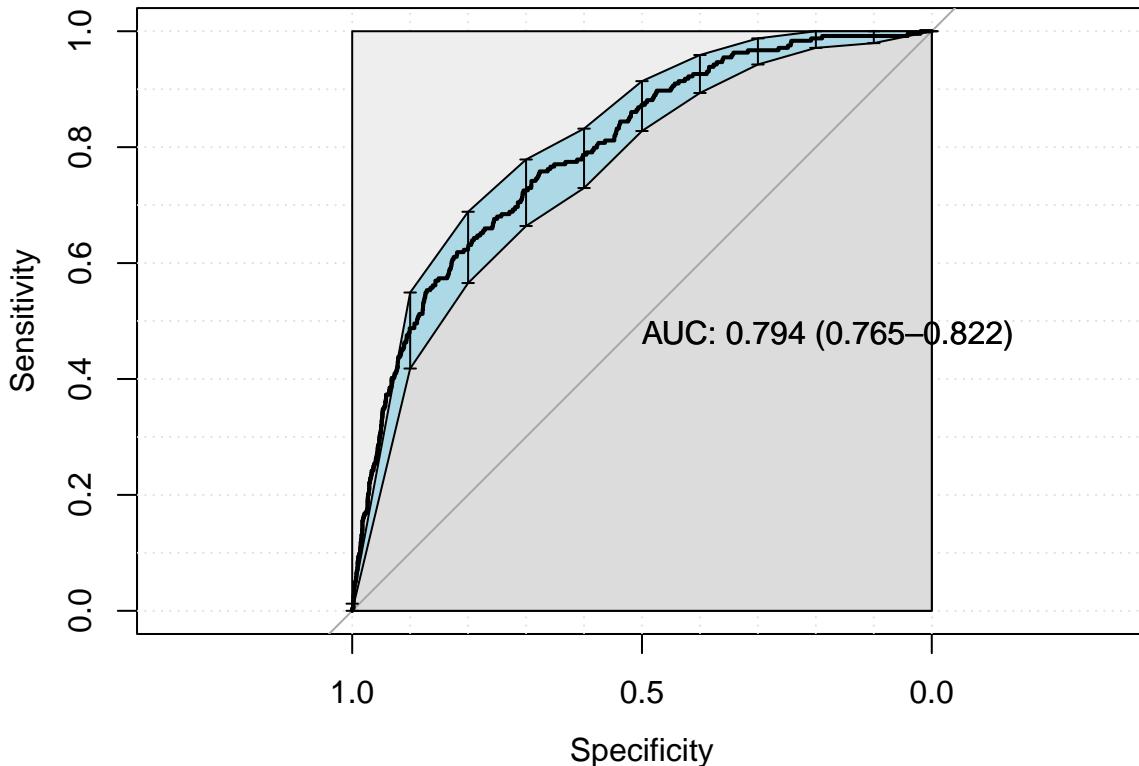
lightgbm_auc <- validation(final_lightgbm_fit, df_test)

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: 0.06"
## Confusion Matrix and Statistics
##
##      reference
## data      0      1

```

```

##      0 3674   93
##      1  812  151
##
##          Accuracy : 0.8087
##             95% CI : (0.7972, 0.8198)
##    No Information Rate : 0.9484
##    P-Value [Acc > NIR] : 1
##
##          Kappa : 0.183
##
## Mcnemar's Test P-Value : <2e-16
##
##          Sensitivity : 0.8190
##          Specificity : 0.6189
##    Pos Pred Value : 0.9753
##    Neg Pred Value : 0.1568
##          Prevalence : 0.9484
##    Detection Rate : 0.7767
## Detection Prevalence : 0.7964
##    Balanced Accuracy : 0.7189
##
##    '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)

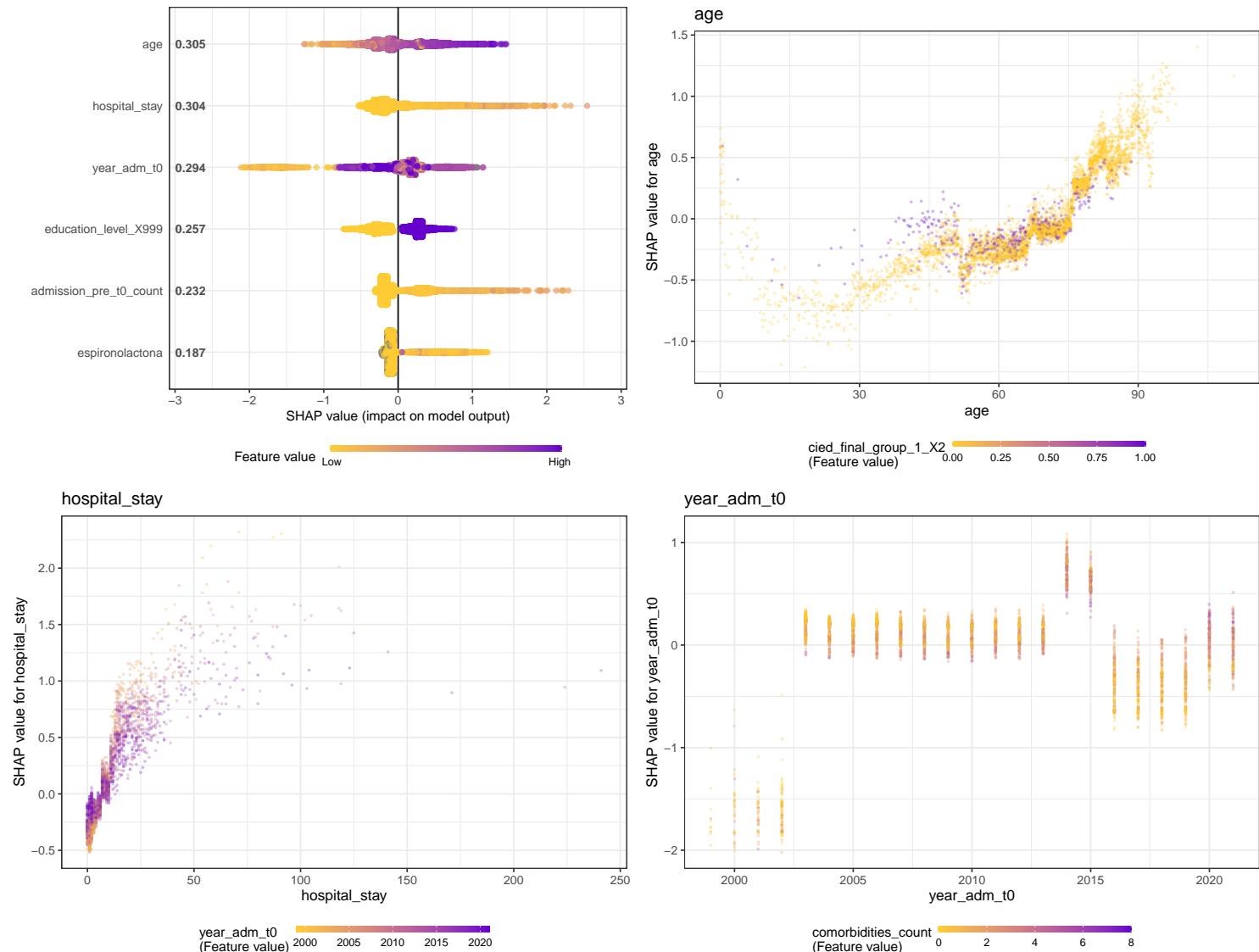
if (plotted < n_plots) {
  print(p)
  plotted <- plotted + 1
}

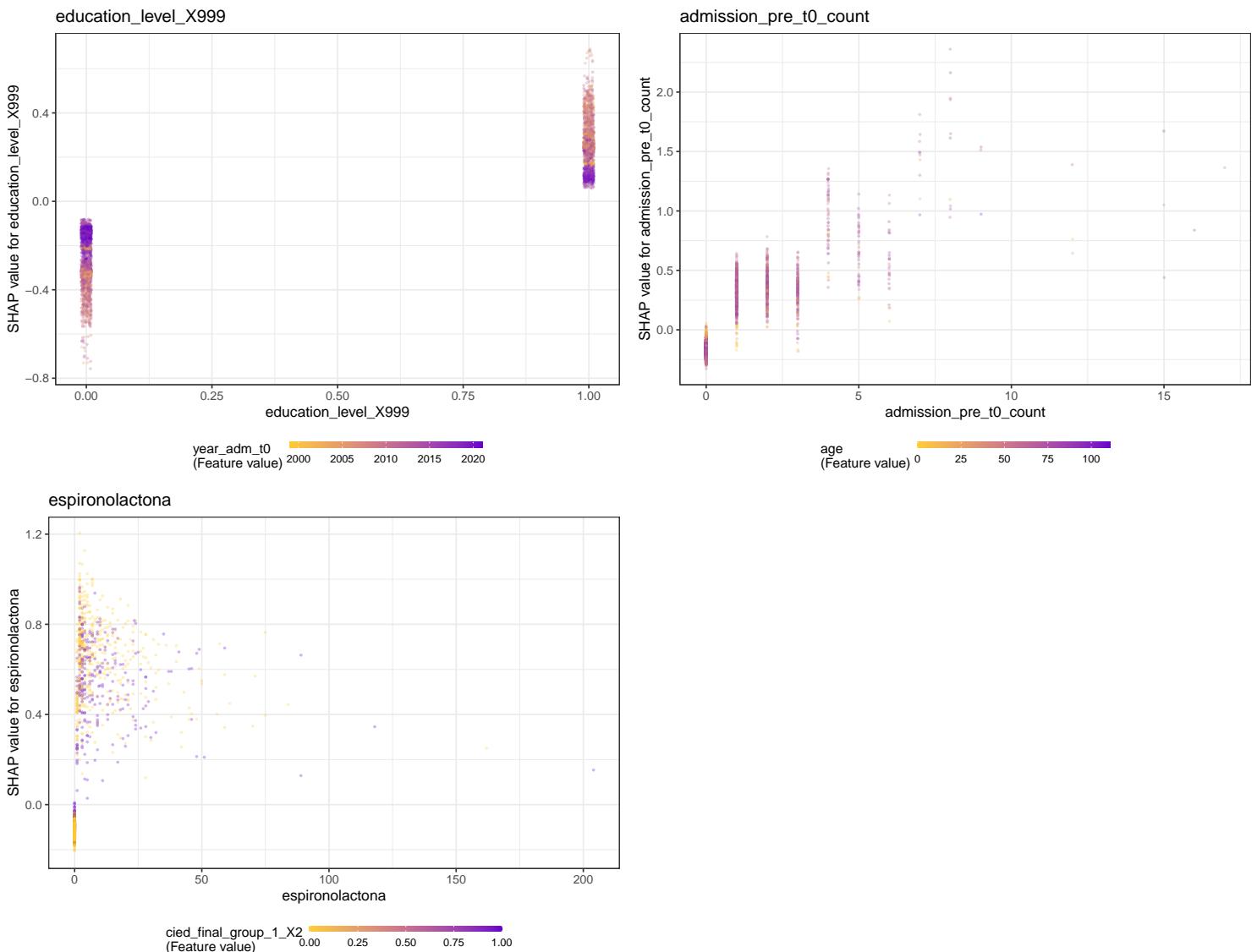
# ggsave(sprintf("./auxiliar/final_model/shap_plots/%s/%s.png", outcome_column, x),
#        plot = p,
#        dpi = 300)
}

```

Warning: Removed 5 rows containing missing values (geom_point).

Warning: Removed 1044 rows containing missing values (geom_point).





```

## $num_iterations
## [1] 95
##
## $learning_rate
## [1] 0.07361728
##
## $max_depth
## [1] 4
##
## $feature_fraction
## [1] 1
##
## $min_data_in_leaf
## [1] 58
##
## $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] 30679
##
## $deterministic
## [1] TRUE
##
## $verbose
## [1] -1
##
## $metric
## list()
##
## $interaction_constraints
## list()
##
## $feature_pre_filter
## [1] FALSE

```

Models Comparison

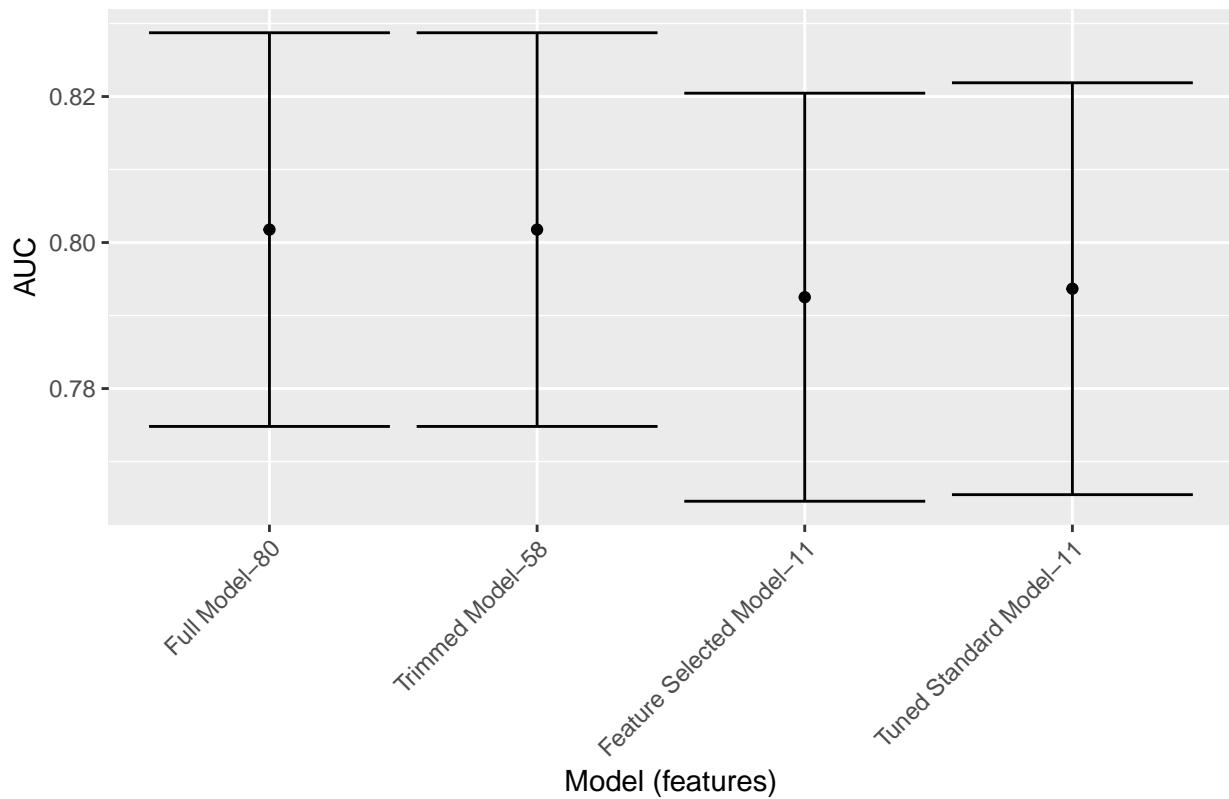
```

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$auc_upper,
  'Tuned Standard Model', standard_results$auc, standard_results$auc_lower, standard_results$auc_upper, length(standard_features)
) %>%
  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))

```

death_2year



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