

Final Model - readmission_30d

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= readmission_30d
## . 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. education_level
2. underlying_heart_disease
3. heart_disease
4. nyha_basal
5. prior_mi
6. heart_failure

7. transplant
8. endocardites
9. hemodialysis
10. comorbidities_count
11. procedure_type_1
12. reop_type_1
13. procedure_type_new
14. cied_final_1
15. cied_final_group_1
16. admission_pre_t0_count
17. admission_pre_t0_180d
18. icu_t0
19. dialysis_t0
20. admission_t0_emergency
21. aco
22. antiarritmico
23. betabloqueador
24. ieca_bra
25. dva
26. digoxina
27. estatina
28. diuretico
29. vasodilatador
30. insuf_cardiaca
31. espironolactona
32. bloq_calcio
33. antiplaquetario_ev
34. insulina
35. anticonvulsivante
36. psicofarmacos
37. antifungico
38. antiviral
39. classe_meds_qtde
40. meds_cardiovasc_qtde
41. meds_antimicrobianos
42. ventilacao_mecanica
43. cec
44. transplante_cardiaco
45. outros_proced_cirurgicos
46. icp
47. intervencao_cv
48. cateterismo
49. eletrofisiologia
50. cateter_venoso_central
51. proced_invasivos_qtde
52. cve_desf
53. transfusao
54. equipe_multiprof
55. holter
56. metodos_graficos_qtde
57. laboratorio
58. cultura
59. analises_clinicas_qtde
60. citologia
61. biopsia
62. histopatologia_qtde
63. angio_rm
64. angio_tc
65. cintilografia
66. ecocardiograma
67. endoscopia

68. flebografia
 69. pet_ct
 70. ultrassom
 71. tomografia
 72. ressonancia
 73. exams_imagem_qtde
 74. bic
 75. mpp
 76. 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: 0.695"
sprintf('Full Model Test AUC: %.3f' ,full_model$auc)

## [1] "Full Model Test AUC: 0.699"

Features with zero importance on the initial model:

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

unimportant_features %>%
  gluedown::md_order()

1. transplant
2. endocardites
3. procedure_type_1
4. dialysis_t0
5. ieca_bra
6. espironolactona
7. antiplaquetario_ev
8. antifungico
9. antiviral
10. cec
11. transplante_cardiaco
12. icp

```

```

13. intervencao_cv
14. cateterismo
15. eletrofisiologia
16. transfusao
17. biopsia
18. histopatologia_qtde
19. angio_rm
20. angio_tc
21. ecocardiograma
22. pet_ct
23. tomografia

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: 0.696"
sprintf('Trimmed Model Test AUC: %.3f' ,trimmed_model$auc)

## [1] "Trimmed Model Test AUC: 0.699"
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] "52"      "-0.0016"  "-6e-04"   "cultura"
## [1] "51"      "-0.0016"  "0"        "reop_type_1"
## [1] "50"      "-2e-04"   "0.0014"   "nyha_basal"
## [1] "49"      "-8e-04"   "-6e-04"   "insulina"
## [1] "48"      "-6e-04"   "1e-04"    "cintilografia"
## [1] "47"      "-5e-04"   "2e-04"    "ressonancia"
## [1] "46"      "-0.0013"  "-8e-04"   "cied_final_group_1"
## [1] "45"      "-0.0017"  "-4e-04"   "cve_desf"
## [1] "44"      "-0.0016"  "1e-04"    "heart_disease"
## [1] "43"      "-0.0015"  "1e-04"    "heart_failure"
## [1] "42"      "-0.0014"  "2e-04"
## [4] "cateter Venoso Central"
## [1] "41"      "-0.0019"  "-6e-04"
## [4] "admission_pre_t0_180d"
## [1] "40"      "-0.003"   "-0.0011"
## [4] "admission_t0_emergency"
## [1] "39"      "-0.0033"  "-4e-04"   "meds_cardiovasc_qtde"
## [1] "38"      "-0.0052"  "-0.0019"  "estatina"
## [1] "37"      "-0.0048"  "4e-04"    "ultrassom"
## [1] "36"      "-0.006"   "-0.0012"  "endoscopia"
## [1] "35"      "-0.0056"  "3e-04"    "flebografia"
## [1] "34"      "-0.0065"  "-8e-04"   "mpp"
## [1] "33"      "-0.0056"  "9e-04"    "psicofarmacos"
## [1] "32"      "-0.0062"  "-6e-04"   "bloq_calcio"
## [1] "31"      "-0.0076"  "-0.0013"   "exames_imagem_qtde"
## [1] "30"      "-0.0069"  "7e-04"    "dva"
## [1] "29"      "-0.0081"  "-0.0012"  "holter"

```

```

## [1] "28"           "-0.0071"          "0.001"           "ventilacao_mecanica"
## [1] "27"           "-0.0073"          "-2e-04"          "laboratorio"
## [1] "26"           "-0.0067"          "6e-04"           "hemodialysis"
## [1] "25"           "-0.0054"          "0.0012"          "citologia"
## [1] "24"           ""                "-0.0053"         "1e-04"
## [4] "outros_proced_cirurgicos"
## [1] "23"           "-0.0038"          "0.0016"          "cied_final_1"
## [1] "23"           ""                "-0.0038"         "0.0023"          "procedure_type_new"
## [1] "22"           "-0.0047"          "-9e-04"          "aco"
## [1] "21"           "-0.0055"          "-8e-04"          "insuf_cardiaca"
## [1] "20"           ""                "-0.0074"         "-0.0019"
## [4] "underlying_heart_disease"
## [1] "20"           "-0.0074"          "0.0022"          "prior_mi"
## [1] "19"           ""                "-0.0058"         "0.0015"
## [4] "metodos_graficos_qtde"
## [1] "18"           "-0.0064"          "-5e-04"          "betabloqueador"
## [1] "18"           ""                "-0.0064"         "0.003"           "comorbidities_count"
## [1] "17"           "-0.0092"          "-0.0028"         "vasodilatador"
## [1] "16"           "-0.0078"          "0.0014"          "equipe_multiprof"
## [1] "15"           ""                "-0.0078"         "0"
## [4] "proced_invasivos_qtde"
## [1] "15"           "-0.0078"          "0.0037"          "anticonvulsivante"
## [1] "15"           "-0.0078"          "0.0037"          "icu_t0"
## [1] "15"           "-0.0078"          "0.0053"          "digoxina"
## [1] "15"           ""                "-0.0078"         "0.0035"          "antiarritmico"
## [1] "15"           "-0.0078"          "0.0034"          "bic"
## [1] "14"           "-0.0087"          "-9e-04"          "diuretico"
## [1] "13"           ""                "-0.0094"         "-7e-04"
## [4] "analises_clinicas_qtde"
## [1] "12"           "-0.01"            "-5e-04"          "education_level"
## [1] "11"           "-0.0087"          "0.0012"          "classe_meds_qtde"
## [1] "11"           ""                "-0.0087"         "0.0035"          "meds_antimicrobianos"
## [1] "11"           ""                "-0.0087"         "0.0052"
## [4] "admission_pre_t0_count"
## [1] "11"           "-0.0087"          "0.0245"          "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	76	0.6953	0.0080	0.0000	0.0000
All unimportant	TRUE	53	0.6963	0.0078	-0.0010	-0.0010
cultura	TRUE	52	0.6969	0.0079	-0.0016	-0.0006
reop_type_1	TRUE	51	0.6969	0.0079	-0.0016	0.0000
nyha_basal	TRUE	50	0.6955	0.0080	-0.0002	0.0014
insulina	TRUE	49	0.6961	0.0078	-0.0008	-0.0006
cintilografia	TRUE	48	0.6960	0.0081	-0.0006	0.0001
ressonancia	TRUE	47	0.6958	0.0081	-0.0005	0.0002
cied_final_group_1	TRUE	46	0.6966	0.0079	-0.0013	-0.0008
cve_desf	TRUE	45	0.6970	0.0080	-0.0017	-0.0004
heart_disease	TRUE	44	0.6969	0.0082	-0.0016	0.0001
heart_failure	TRUE	43	0.6969	0.0082	-0.0015	0.0001
cateter Venoso Central	TRUE	42	0.6967	0.0082	-0.0014	0.0002
admission_pre_t0_180d	TRUE	41	0.6973	0.0083	-0.0019	-0.0006
admission_t0_emergency	TRUE	40	0.6983	0.0083	-0.0030	-0.0011
meds_cardiovasc_qtde	TRUE	39	0.6987	0.0084	-0.0033	-0.0004
estatina	TRUE	38	0.7006	0.0085	-0.0052	-0.0019

Table 1: (continued)

Tested Feature	Dropped	Features	CV AUC	CV AUC Std Error	Total AUC Loss	Instant AUC Loss
ultrassom	TRUE	37	0.7002	0.0083	-0.0048	0.0004
endoscopia	TRUE	36	0.7013	0.0080	-0.0060	-0.0012
flebografia	TRUE	35	0.7010	0.0081	-0.0056	0.0003
mpp	TRUE	34	0.7018	0.0084	-0.0065	-0.0008
psicofarmacos	TRUE	33	0.7010	0.0082	-0.0056	0.0009
bloq_calcio	TRUE	32	0.7016	0.0080	-0.0062	-0.0006
exames_imagem_qtde	TRUE	31	0.7029	0.0082	-0.0076	-0.0013
dva	TRUE	30	0.7023	0.0083	-0.0069	0.0007
holter	TRUE	29	0.7034	0.0084	-0.0081	-0.0012
ventilacao_mecanica	TRUE	28	0.7025	0.0083	-0.0071	0.0010
laboratorio	TRUE	27	0.7027	0.0081	-0.0073	-0.0002
hemodialysis	TRUE	26	0.7020	0.0083	-0.0067	0.0006
citologia	TRUE	25	0.7008	0.0082	-0.0054	0.0012
outros_proced_cirurgicos	TRUE	24	0.7007	0.0082	-0.0053	0.0001
cied_final_1	TRUE	23	0.6991	0.0082	-0.0038	0.0016
procedure_type_new	FALSE	22	0.6968	0.0084	-0.0038	0.0023
aco	TRUE	22	0.7001	0.0082	-0.0047	-0.0009
insuf_cardiaca	TRUE	21	0.7009	0.0083	-0.0055	-0.0008
underlying_heart_disease	TRUE	20	0.7027	0.0086	-0.0074	-0.0019
prior_mi	FALSE	19	0.7005	0.0086	-0.0074	0.0022
metodos_graficos_qtde	TRUE	19	0.7012	0.0090	-0.0058	0.0015
betabloqueador	TRUE	18	0.7017	0.0091	-0.0064	-0.0005
comorbidities_count	FALSE	17	0.6987	0.0089	-0.0064	0.0030
vasodilatador	TRUE	17	0.7046	0.0092	-0.0092	-0.0028
equipe_multiprof	TRUE	16	0.7032	0.0092	-0.0078	0.0014
proced_invasivos_qtde	TRUE	15	0.7031	0.0093	-0.0078	0.0000
anticonvulsivante	FALSE	14	0.6994	0.0095	-0.0078	0.0037
icu_t0	FALSE	14	0.6994	0.0095	-0.0078	0.0037
digoxina	FALSE	14	0.6979	0.0094	-0.0078	0.0053
antiarritmico	FALSE	14	0.6996	0.0095	-0.0078	0.0035
bic	FALSE	14	0.6997	0.0089	-0.0078	0.0034
diuretico	TRUE	14	0.7041	0.0090	-0.0087	-0.0009
analises_clinicas_qtde	TRUE	13	0.7048	0.0087	-0.0094	-0.0007
education_level	TRUE	12	0.7053	0.0092	-0.0100	-0.0005
classe_meds_qtde	TRUE	11	0.7041	0.0086	-0.0087	0.0012
meds_antimicrobianos	FALSE	10	0.7006	0.0085	-0.0087	0.0035
admission_pre_t0_count	FALSE	10	0.6989	0.0082	-0.0087	0.0052
hospital_stay	FALSE	10	0.6796	0.0081	-0.0087	0.0245

```

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: 0.704"
sprintf('Selected Model Test AUC: %.3f', feature_selected_model$auc)

## [1] "Selected Model Test AUC: 0.690"

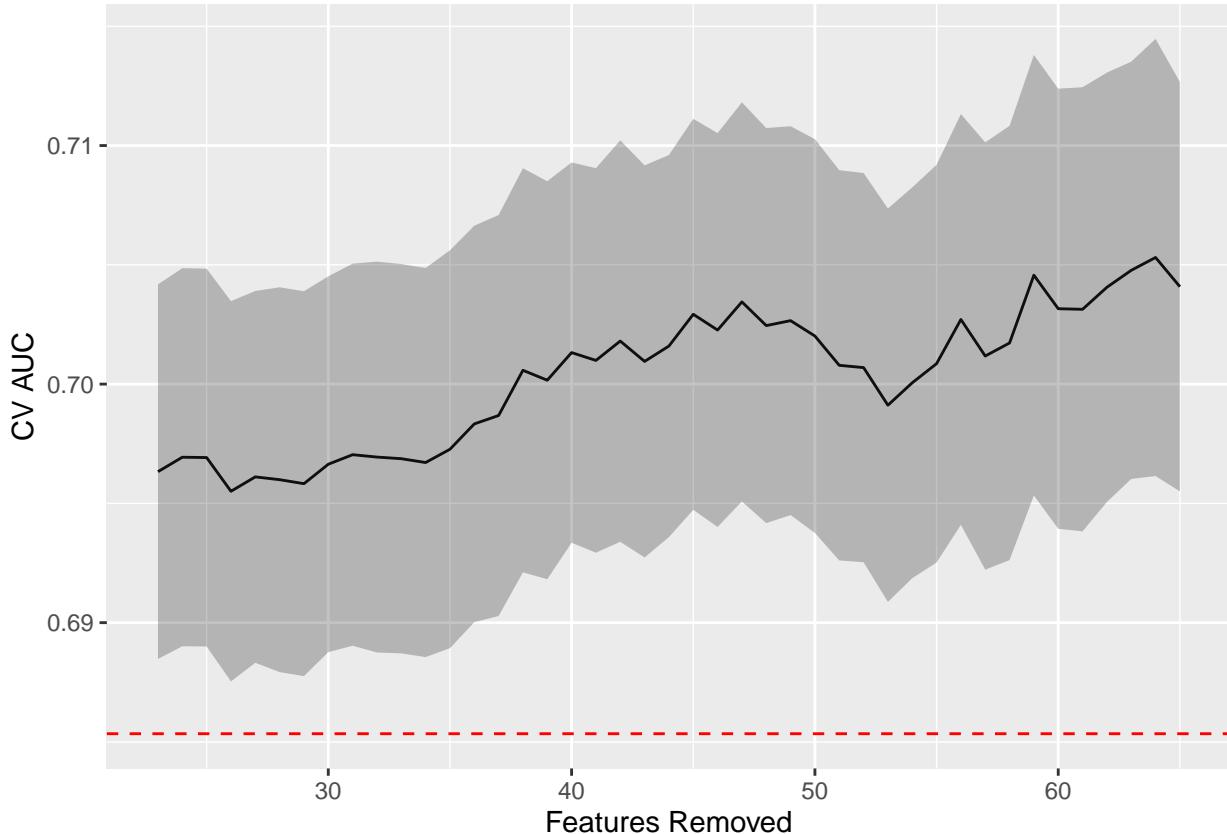
```

```

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. digoxina
4. meds_antimicrobianos
5. bic
6. icu_t0
7. antiaritmico
8. anticonvulsivante
9. comorbidities_count
10. procedure_type_new
11. prior_mi

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)
```

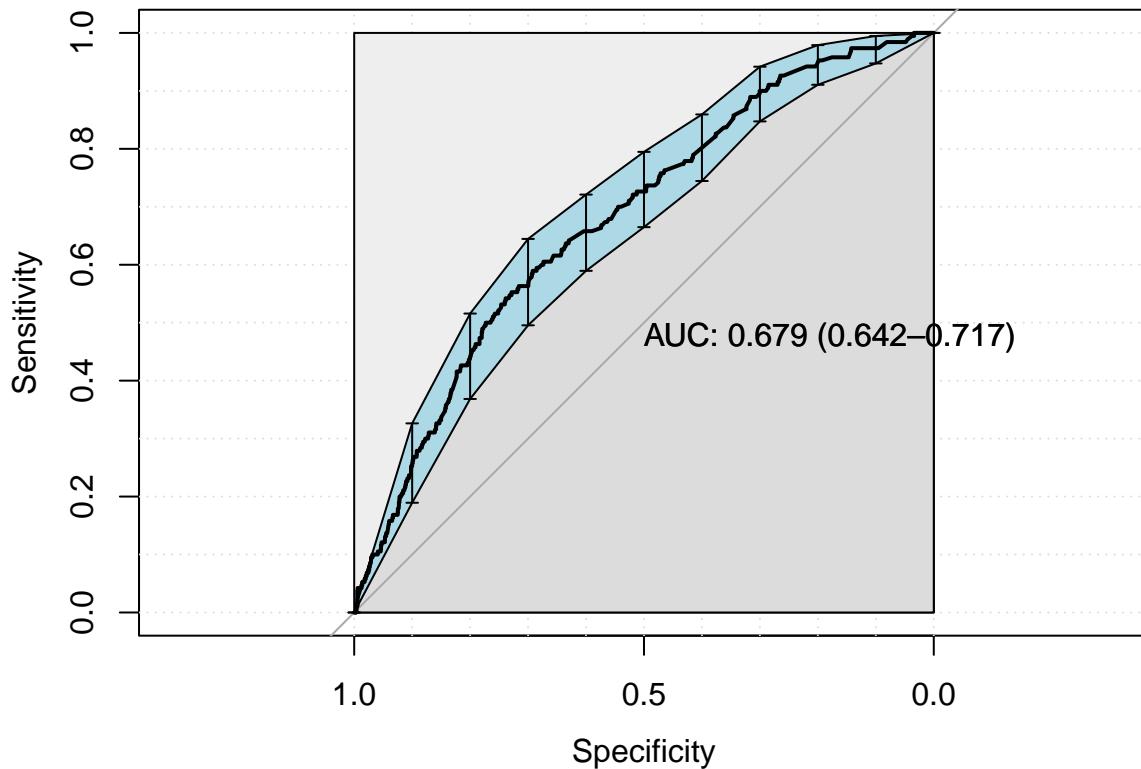
```

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 3142    78
##   1 1398   112
##
##                  Accuracy : 0.6879
##                  95% CI : (0.6745, 0.7011)
##      No Information Rate : 0.9598
##      P-Value [Acc > NIR] : 1
##
##                  Kappa : 0.065
##
## Mcnemar's Test P-Value : <2e-16
##
##      Sensitivity : 0.69207
##      Specificity : 0.58947

```

```

##          Pos Pred Value : 0.97578
##          Neg Pred Value : 0.07417
##          Prevalence : 0.95983
##          Detection Rate : 0.66427
## Detection Prevalence : 0.68076
##          Balanced Accuracy : 0.64077
##
##          '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)

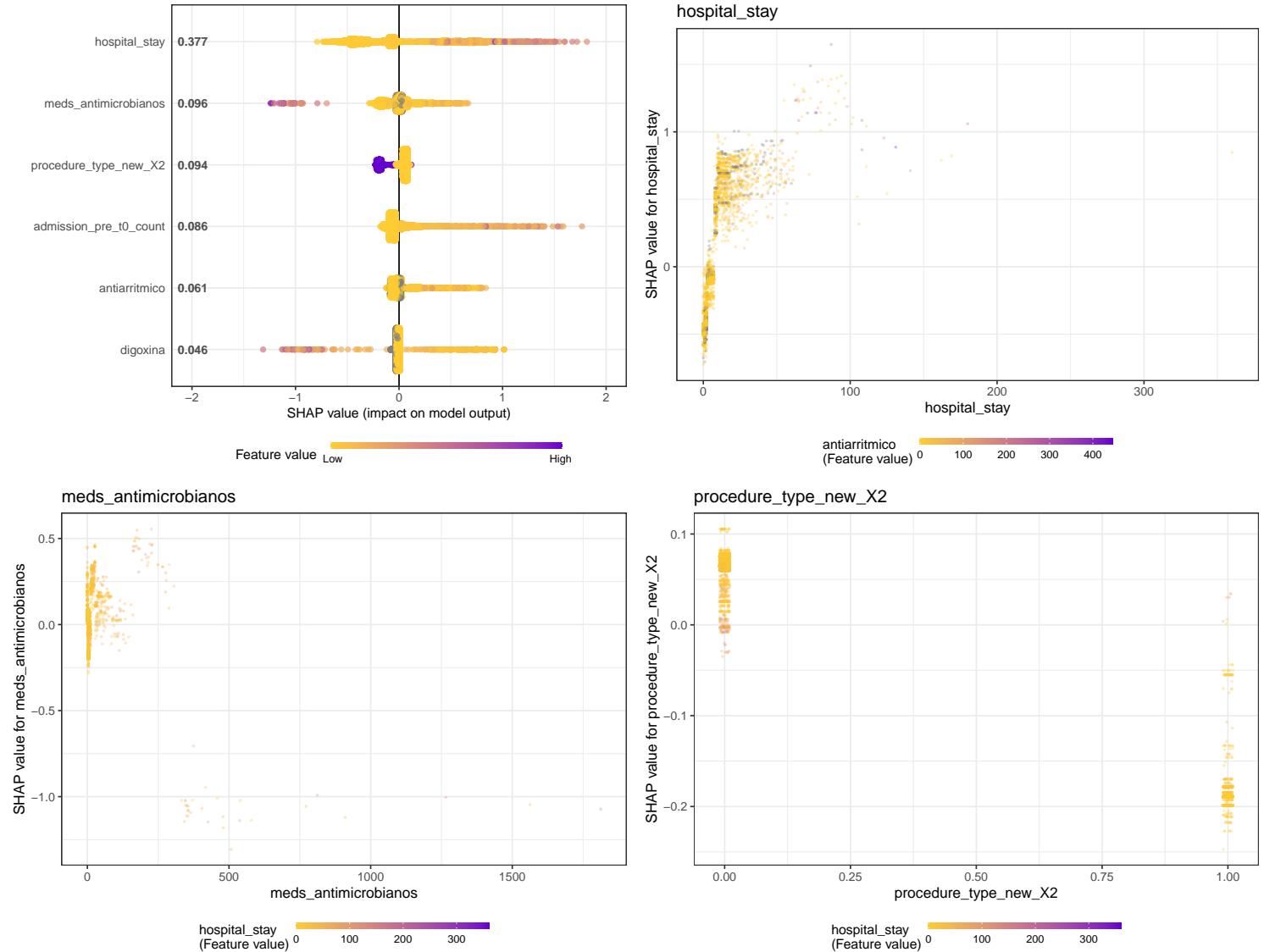
  if (plotted < n_plots) {
    print(p)
    plotted <- plotted + 1
  }
}
```

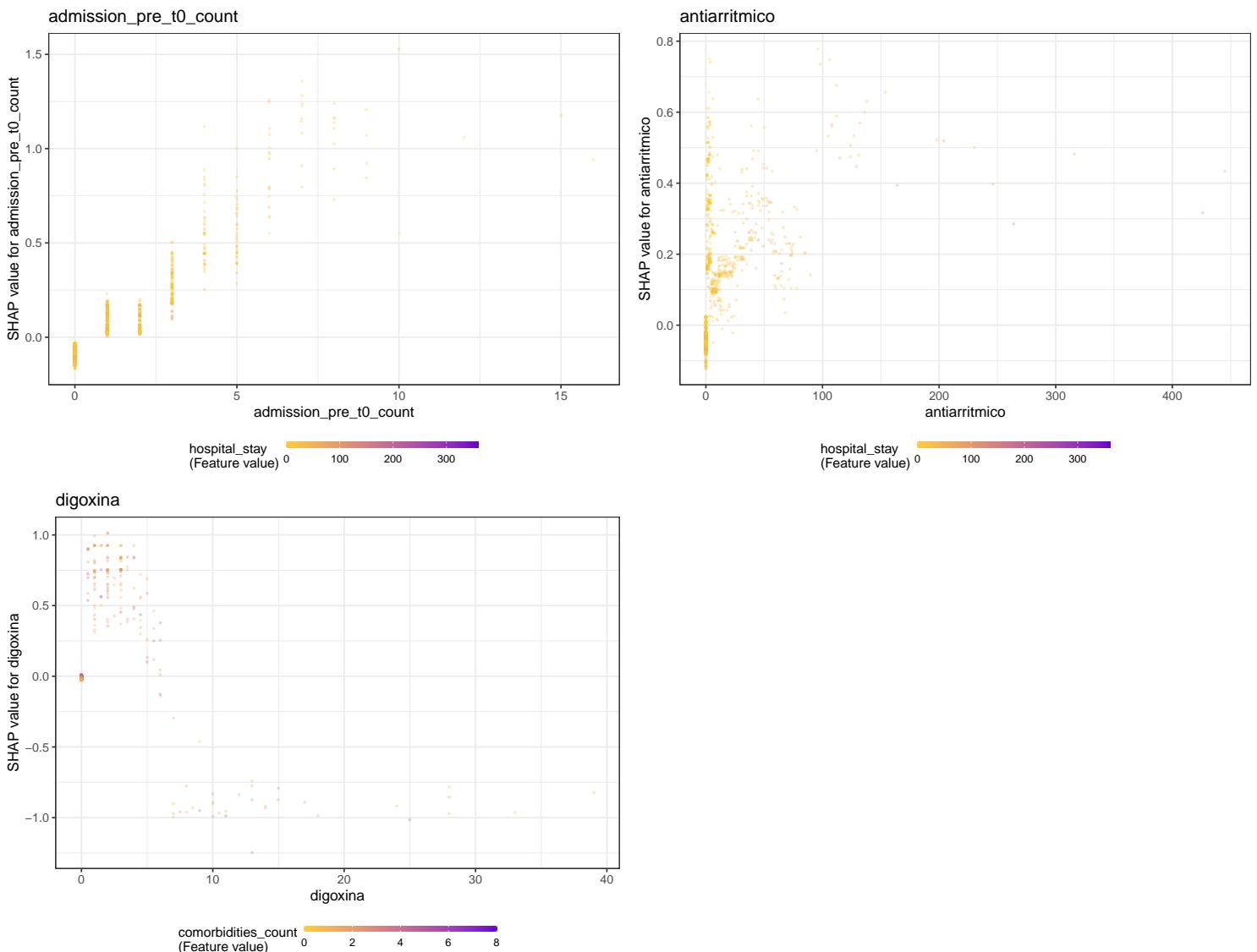
```

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

## Warning: Removed 1050 rows containing missing values (geom_point).
## Removed 1050 rows containing missing values (geom_point).
## Removed 1050 rows containing missing values (geom_point).

```





```

## $num_iterations
## [1] 94
##
## $learning_rate
## [1] 0.06873269
##
## $max_depth
## [1] 3
##
## $feature_fraction
## [1] 1
##
## $min_data_in_leaf
## [1] 52
##
## $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] 31044
##
## $deterministic
## [1] TRUE
##
## $verbose
## [1] -1
##
## $metric
## list()
##
## $interaction_constraints
## list()
##
## $feature_pre_filter
## [1] FALSE

```

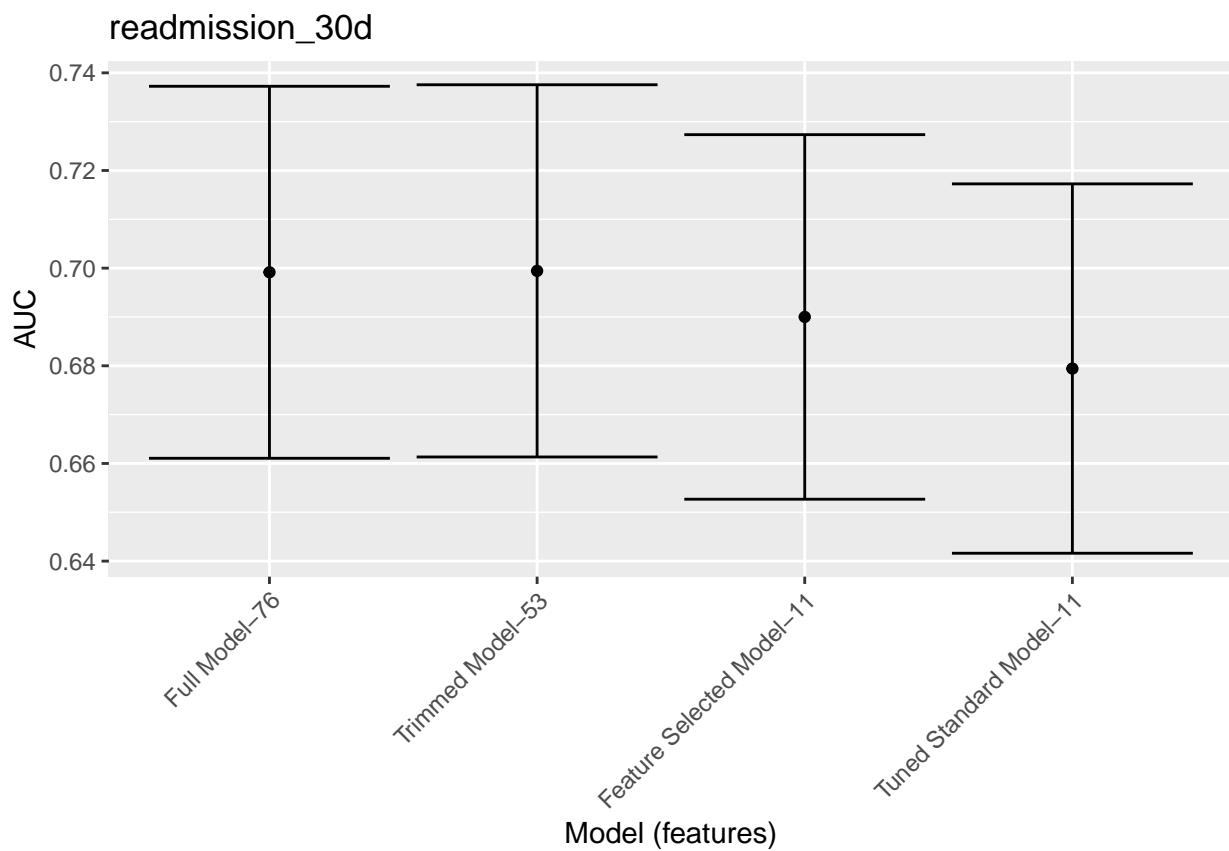
Models Comparison

```

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))

```



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