Final Model - death_30days

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
```

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))

if (is.null(features_list)) {
  features = eligible_features
} else {
  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. valvopathy

12. diabetes
13. renal_failure
14. hemodialysis
15. cancer
16. comorbidities_count
17. procedure_type_1
18. reop_type_1
19. procedure_type_new
20. cied_final_1
21. cied_final_group_1
22. admission_pre_t0_count
23. admission_pre_t0_180d
24. year_adm_t0
25. icu_t0
26. antiarritmico
27. antihipertensivo
28. betabloqueador
29. dva
30. diuretico
31. vasodilatador
32. espironolactona
33. antiplaquetario_ev
34. insulina
35. psicofarmacos
36. antifungico
37. classe_meds_qtde
38. meds_cardiovasc_qtde
39. meds_antimicrobianos
40. vni
41. ventilacao_mecanica
42. intervencao_cv
43. cateter_venoso_central
44. proced_invasivos_qtde
45. transfusao
46. interconsulta
47. equipe_multiprof
48. holter
49. metodos_graficos_qtde
50. laboratorio
51. cultura
52. analises_clinicas_qtde
53. citologia
54. histopatologia_qtde
55. angio_tc
56. angiografia
57. cintilografia
58. ecocardiograma
59. flebografia
60. ultrassom
61. tomografia
62. ressonancia
63. exames_imagem_qtde
64. bic
65. 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: 0.753"
```

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

```
## [1] "Full Model Test AUC: 0.794"
```

```
# full_model$importance %>%
#   filter(str_detect(Feature, 'education'))
#
# full_model$importance %>%
#   filter(str_detect(Feature, 'education')) %>%
#   summarise(across(where(is.numeric), ~ sum(.x, na.rm = TRUE)))
#
# full_model$importance %>%
#   separate(Feature, c("Feature","value"), "__") %>%
#   group_by(Feature) %>%
#   summarise(Gain = sum(Gain),
#             Cover = sum(Cover),
#             Frequency = sum(Frequency))
```

Features with zero importance on the initial model:

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

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

1. sex
2. hypertension
3. prior_mi
4. heart_failure
5. valvopathy
6. cancer
7. admission_pre_t0_180d
8. betabloqueador
9. antiplaquetario_ev
10. vni
11. ventilacao_mecanica
12. intervencao_cv

13. transfusao
14. interconsulta
15. histopatologia_qtde
16. angio_tc
17. flebografia
18. ultrassom

```
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: 0.757"
```

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

```
## [1] "Trimmed Model Test AUC: 0.794"
```

```
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] "46"                "-0.0041"           "0"
## [4] "procedure_type_1"
## [1] "45"        "-0.009"    "-0.005"    "insulina"
## [1] "44"                "-0.0106"           "-0.0015"
## [4] "antihipertensivo"
## [1] "43"                  "-0.0168"             "-0.0063"
## [4] "procedure_type_new"
## [1] "42"             "-0.0196"        "-0.0027"        "heart_disease"
## [1] "41"           "-0.02"        "-4e-04"       "antifungico"
## [1] "40"             "-0.0229"        "-0.003"         "cintilografia"
## [1] "39"                      "-0.025"
## [3] "-0.002"                  "cateter_venoso_central"
## [1] "38"          "-0.0279"     "-0.0029"     "tomografia"
## [1] "38"           "-0.0279"      "0.0025"       "reop_type_1"
## [1] "38"            "-0.0279"       "0.0025"        "cied_final_1"
## [1] "37"                    "-0.03"               "-0.0021"
## [4] "comorbidities_count"
## [1] "37"              "-0.03"           "0.0043"          "ecocardiograma"
## [1] "36"       "-0.0317" "-0.0016" "dva"
## [1] "35"         "-0.0359"    "-0.0042"    "diuretico"
## [1] "34"             "-0.0405"        "-0.0047"        "antiarritmico"
## [1] "33"                      "-0.0425"               "-0.002"
## [4] "proced_invasivos_qtde"
## [1] "33"                  "-0.0425"             "0.0034"
## [4] "exames_imagem_qtde"
## [1] "32"        "-0.0442"  "-0.0017"  "diabetes"
## [1] "31"           "-0.0464"      "-0.0022"      "ressonancia"
## [1] "31"             "-0.0464"        "0.0026"         "renal_failure"
## [1] "31"         "-0.0464"    "0.0057"     "citologia"
## [1] "30"            "-0.052"        "-0.0056"       "hemodialysis"
## [1] "29"                  "-0.05"             "0.002"
## [4] "cied_final_group_1"
## [1] "28"          "-0.0514"     "-0.0014"     "nyha_basal"
```

```
## [1] "27"                "-0.0518"          "-4e-04"
## [4] "equipe_multiprof"
## [1] "27"      "-0.0518" "0.0042"  "af"
## [1] "26"      "-0.0525" "-7e-04"  "holter"
## [1] "26"                    "-0.0525"              "0.0058"
## [4] "meds_cardiovasc_qtde"
## [1] "25"                        "-0.0668"
## [3] "-0.0143"                   "underlying_heart_disease"
## [1] "25"              "-0.0668"         "0.0089"          "espironolactona"
## [1] "25"      "-0.0668" "0.0041"  "bic"
## [1] "24"      "-0.0667" "1e-04"   "cultura"
## [1] "23"      "-0.0739" "-0.0071" "icu_t0"
## [1] "22"          "-0.0732"     "6e-04"       "laboratorio"
## [1] "21"                      "-0.0801"               "-0.0069"
## [4] "metodos_graficos_qtde"
## [1] "20"                "-0.0823"          "-0.0022"
## [4] "classe_meds_qtde"
## [1] "19"                    "-0.0821"              "2e-04"
## [4] "meds_antimicrobianos"
## [1] "19"          "-0.0821"     "0.0231"      "angiografia"
## [1] "19"          "-0.0821"     "0.0079"      "year_adm_t0"
## [1] "18"            "-0.0811"       "0.001"         "vasodilatador"
## [1] "18"            "-0.0811"       "0.0027"        "psicofarmacos"
## [1] "18"              "-0.0811"         "0.0182"          "education_level"
## [1] "18"                     "-0.0811"
## [3] "0.0246"                 "admission_pre_t0_count"
## [1] "18"                     "-0.0811"
## [3] "0.002"                  "analises_clinicas_qtde"
## [1] "17"            "-0.088"        "-0.0069"       "hospital_stay"
## [1] "17"     "-0.088" "0.025"  "age"
```

```
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	65	0.7528	0.0308	0.0000	0.0000
All unimportant	TRUE	47	0.7568	0.0310	-0.0041	-0.0041
procedure_type_1	TRUE	46	0.7568	0.0310	-0.0041	0.0000
insulina	TRUE	45	0.7618	0.0319	-0.0090	-0.0050
antihipertensivo	TRUE	44	0.7633	0.0326	-0.0106	-0.0015
procedure_type_new	TRUE	43	0.7696	0.0314	-0.0168	-0.0063
heart_disease	TRUE	42	0.7723	0.0322	-0.0196	-0.0027
antifungico	TRUE	41	0.7727	0.0324	-0.0200	-0.0004
cintilografia	TRUE	40	0.7757	0.0332	-0.0229	-0.0030
cateter_venoso_central	TRUE	39	0.7778	0.0325	-0.0250	-0.0020
tomografia	TRUE	38	0.7807	0.0317	-0.0279	-0.0029
reop_type_1	FALSE	37	0.7782	0.0312	-0.0279	0.0025
cied_final_1	FALSE	37	0.7782	0.0311	-0.0279	0.0025
comorbidities_count	TRUE	37	0.7828	0.0320	-0.0300	-0.0021
ecocardiograma	FALSE	36	0.7785	0.0305	-0.0300	0.0043
dva	TRUE	36	0.7844	0.0313	-0.0317	-0.0016
diuretico	TRUE	35	0.7887	0.0320	-0.0359	-0.0042
antiarritmico	TRUE	34	0.7933	0.0311	-0.0405	-0.0047
proced_invasivos_qtde	TRUE	33	0.7953	0.0317	-0.0425	-0.0020
exames_imagem_qtde	FALSE	32	0.7919	0.0334	-0.0425	0.0034
diabetes	TRUE	32	0.7970	0.0308	-0.0442	-0.0017
ressonancia	TRUE	31	0.7992	0.0313	-0.0464	-0.0022

Table 1: *(continued)*

Tested Feature	Dropped	Features	CV AUC	CV AUC Std Error	Total AUC Loss	Instant AUC Loss
renal_failure	FALSE	30	0.7966	0.0312	-0.0464	0.0026
citologia	FALSE	30	0.7935	0.0322	-0.0464	0.0057
hemodialysis	TRUE	30	0.8047	0.0308	-0.0520	-0.0056
cied_final_group_1	TRUE	29	0.8028	0.0317	-0.0500	0.0020
nyha_basal	TRUE	28	0.8042	0.0305	-0.0514	-0.0014
equipe_multiprof	TRUE	27	0.8046	0.0310	-0.0518	-0.0004
af	FALSE	26	0.8004	0.0309	-0.0518	0.0042
holter	TRUE	26	0.8053	0.0305	-0.0525	-0.0007
meds_cardiovasc_qtde	FALSE	25	0.7995	0.0292	-0.0525	0.0058
underlying_heart_disease	TRUE	25	0.8196	0.0268	-0.0668	-0.0143
espironolactona	FALSE	24	0.8107	0.0259	-0.0668	0.0089
bic	FALSE	24	0.8156	0.0270	-0.0668	0.0041
cultura	TRUE	24	0.8195	0.0279	-0.0667	0.0001
icu_t0	TRUE	23	0.8267	0.0252	-0.0739	-0.0071
laboratorio	TRUE	22	0.8260	0.0263	-0.0732	0.0006
metodos_graficos_qtde	TRUE	21	0.8329	0.0219	-0.0801	-0.0069
classe_meds_qtde	TRUE	20	0.8351	0.0263	-0.0823	-0.0022
meds_antimicrobianos	TRUE	19	0.8349	0.0253	-0.0821	0.0002
angiografia	FALSE	18	0.8118	0.0226	-0.0821	0.0231
year_adm_t0	FALSE	18	0.8270	0.0207	-0.0821	0.0079
vasodilatador	TRUE	18	0.8339	0.0287	-0.0811	0.0010
psicofarmacos	FALSE	17	0.8312	0.0300	-0.0811	0.0027
education_level	FALSE	17	0.8156	0.0261	-0.0811	0.0182
admission_pre_t0_count	FALSE	17	0.8093	0.0262	-0.0811	0.0246
analises_clinicas_qtde	FALSE	17	0.8319	0.0276	-0.0811	0.0020
hospital_stay	TRUE	17	0.8408	0.0278	-0.0880	-0.0069
age	FALSE	16	0.8158	0.0264	-0.0880	0.0250

```
if (exists('last_feature_dropped')) {
  selected_features <- c(current_features, last_feature_droped)
} else {
  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)
```

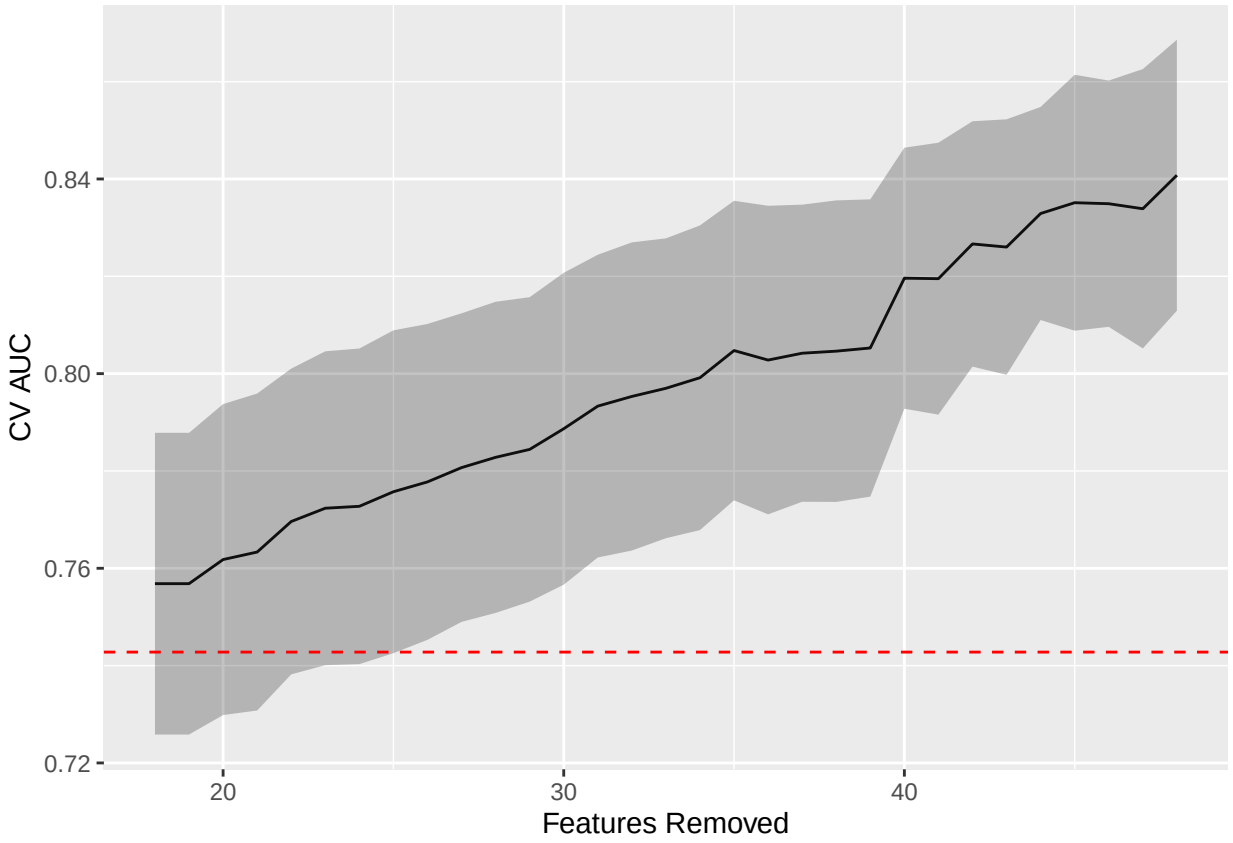
```
## [1] "Selected Model CV Train AUC: 0.841"
```

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

```
## [1] "Selected Model Test AUC: 0.763"
```

```
 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:

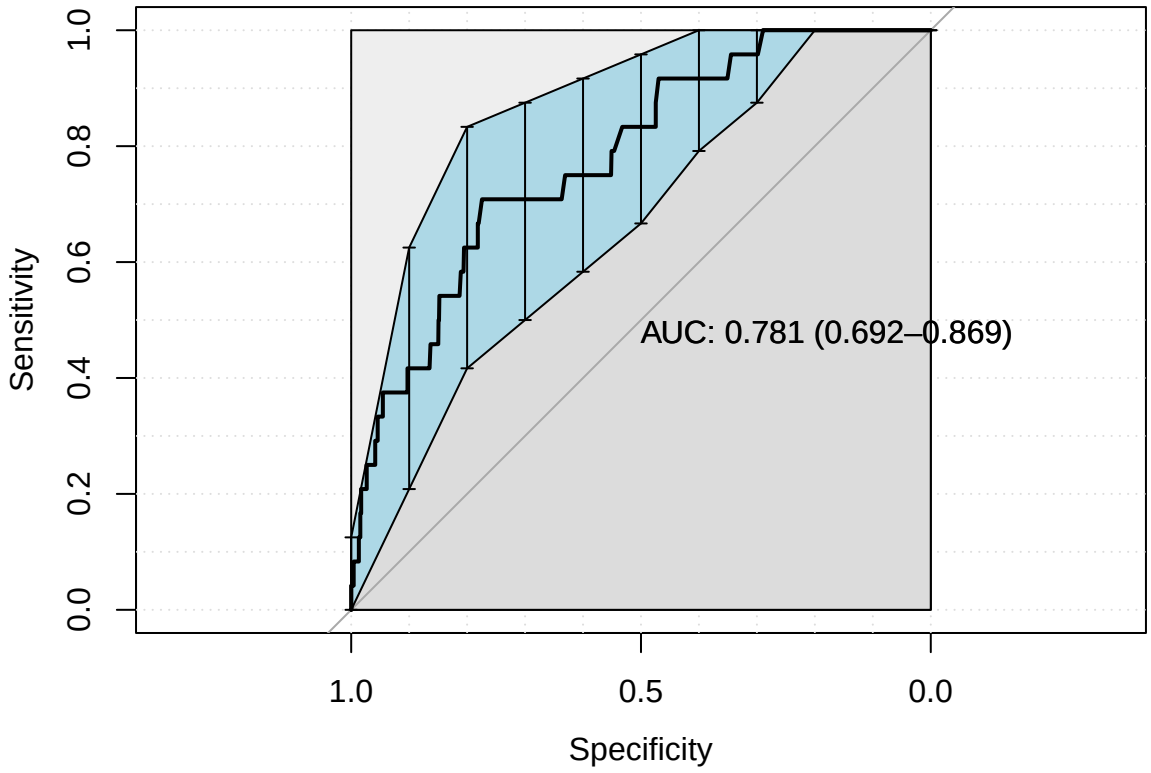
```
gluedown::md_order(selected_features, seq = TRUE, pad = TRUE)
```

1. age
2. admission_pre_t0_count
3. year_adm_t0
4. education_level
5. angiografia
6. psicofarmacos
7. analises_clinicas_qtde
8. bic
9. meds_cardiovasc_qtde
10. renal_failure
11. espironolactona
12. exames_imagem_qtde
13. citologia
14. af
15. ecocardiograma
16. cied_final_1
17. reop_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(),
    # loss_reduction = tune(),
    sample_size = 1.0
  ) %>%
    set_engine("lightgbm") %>%
    set_mode("classification")

  lightgbm_grid <- grid_latin_hypercube(
    trees(range = c(50L, 300L)),
    min_n(),
    tree_depth(),
    learn_rate(range = c(0.01, 0.2), trans = NULL),
    # loss_reduction(),
    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)
```



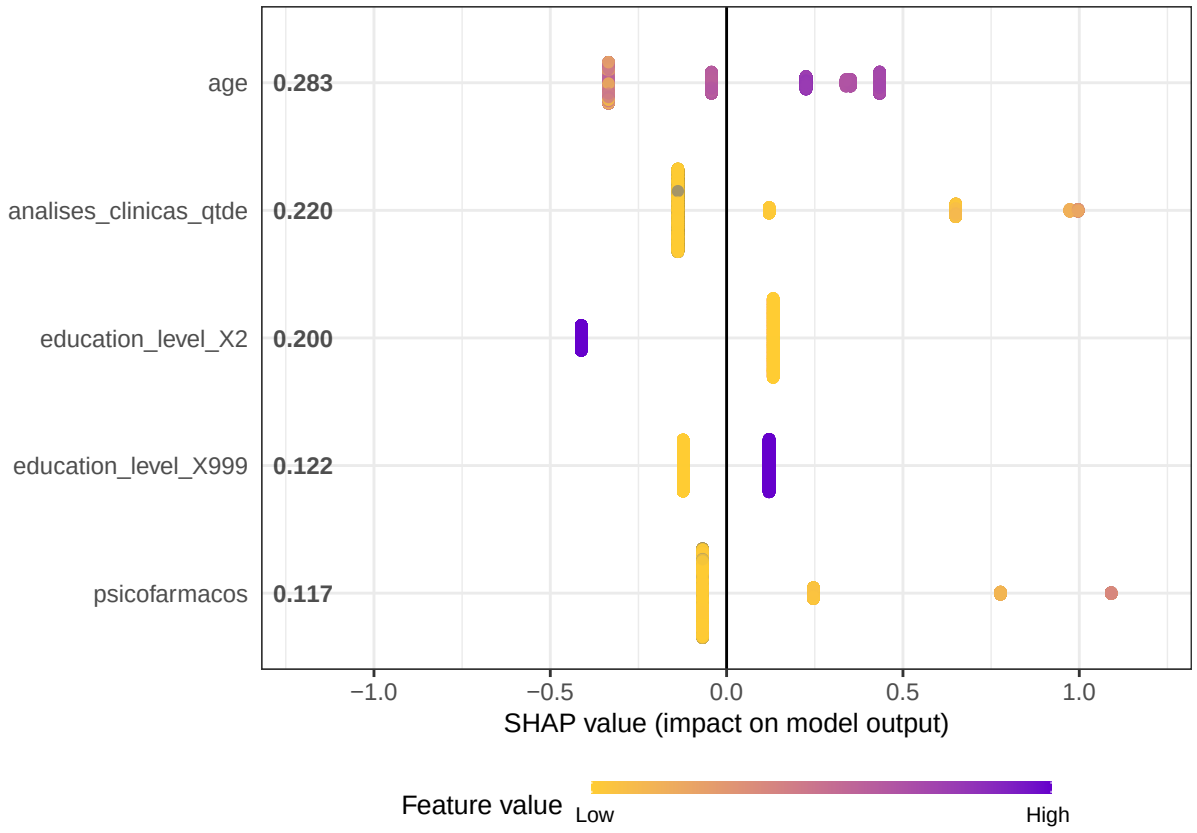

```
## [1] "Optimal Threshold: 0.00"
## Confusion Matrix and Statistics
##
##     reference
## data    0    1
##    0 3644    7
##    1 1062   17
##
##                Accuracy : 0.774
##                  95% CI : (0.7618, 0.7858)
##     No Information Rate : 0.9949
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0211
##
##  Mcnemar's Test P-Value : <2e-16
##
```

```
##             Sensitivity : 0.77433
##             Specificity : 0.70833
##          Pos Pred Value : 0.99808
##          Neg Pred Value : 0.01576
##              Prevalence : 0.99493
##          Detection Rate : 0.77040
##    Detection Prevalence : 0.77188
##       Balanced Accuracy : 0.74133
##
##        'Positive' Class : 0
##
```

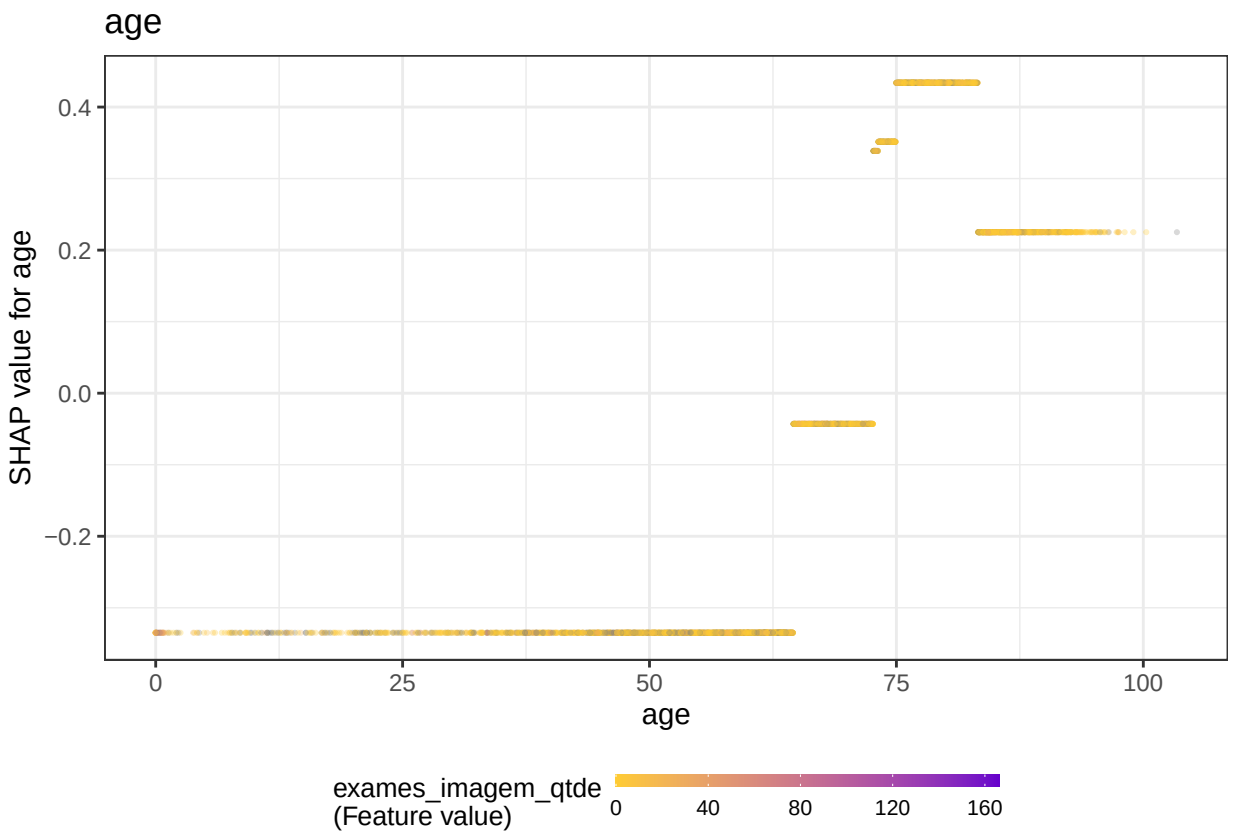
```r
final_lightgbm_fit <- standard_results$fit
lightgbm_parameters <- standard_results$parameters

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

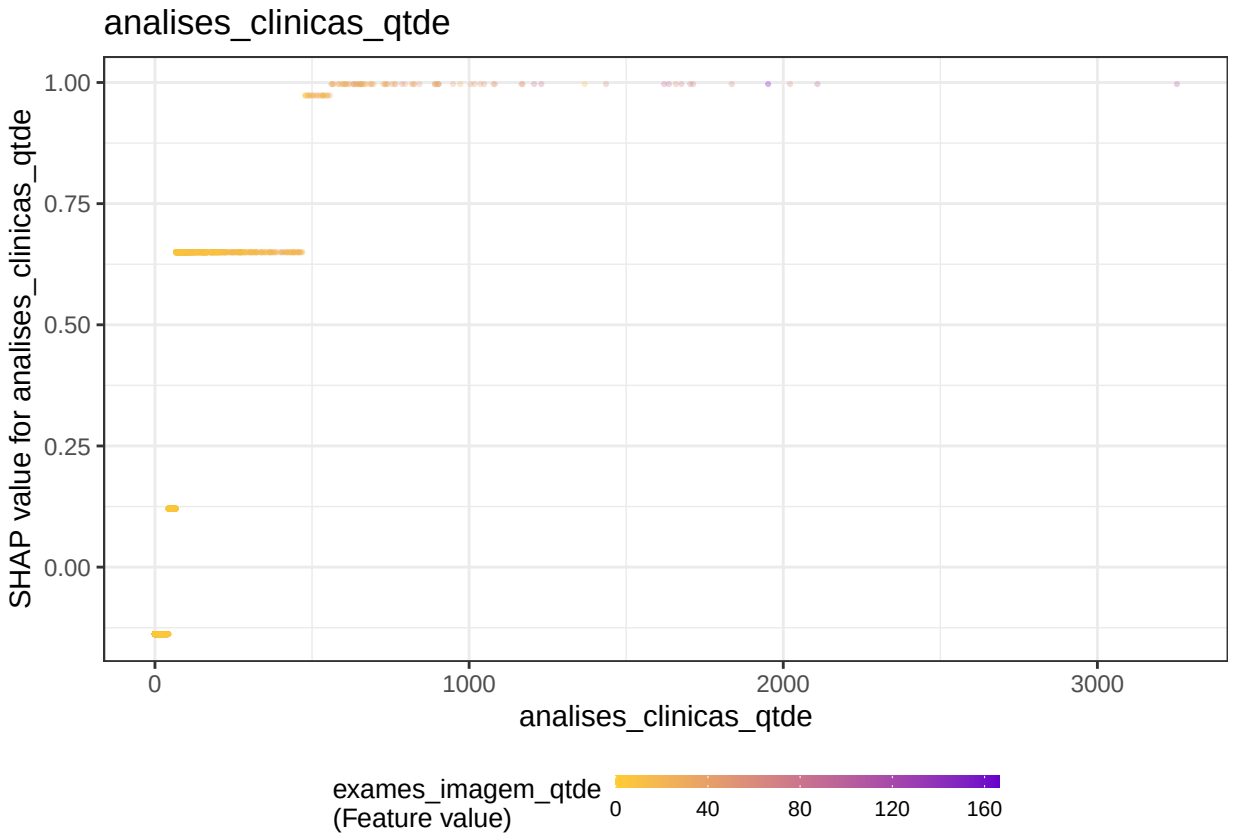
```r
lightgbm_model <- parsnip::extract_fit_engine(final_lightgbm_fit)

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(5, length(selected_features))

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

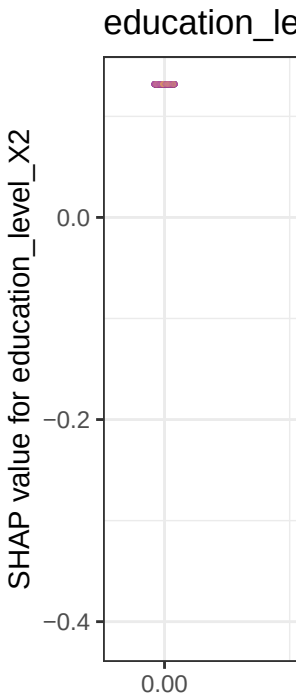


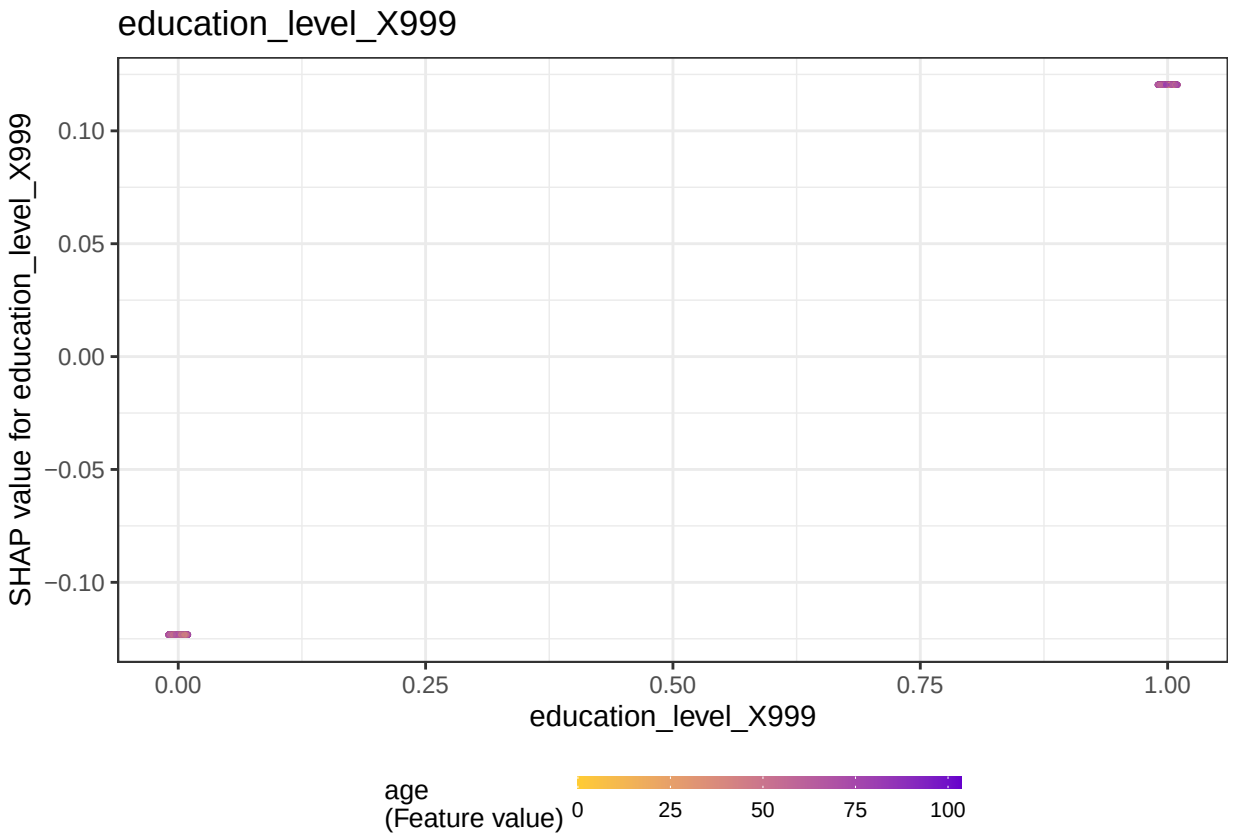

```
shap <- shap.prep(lightgbm_model, X_train = matrix_test)

for (x in shap.importance(shap, names_only = TRUE)[1:n_plots]) {
  p <- shap.plot.dependence(
    shap,
    x = x,
    color_feature = "auto",
    smooth = FALSE,
    jitter_width = 0.01,
    alpha = 0.3
  ) +
    labs(title = x)
  print(p)
}
```



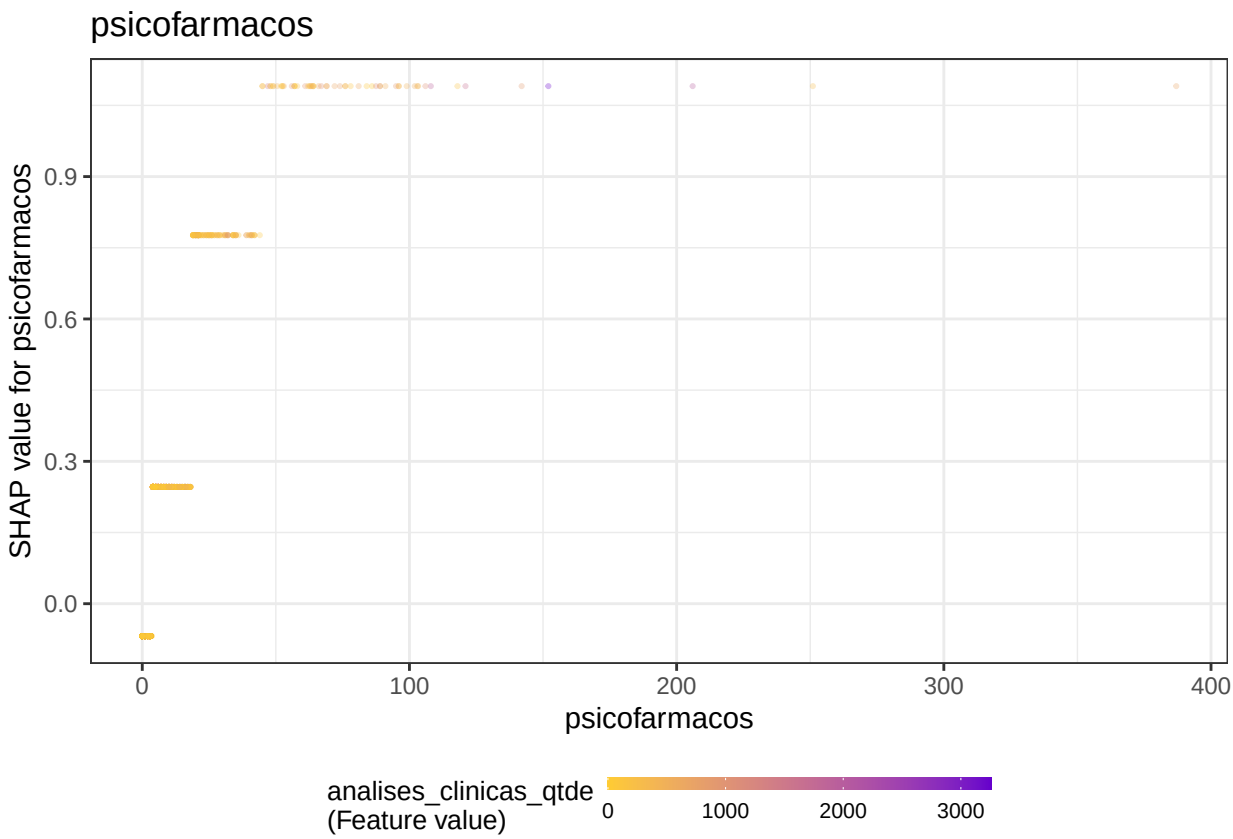

```
## Warning: Removed 822 rows containing missing values (geom_point).
```








```
## Warning: Removed 1055 rows containing missing values (geom_point).
```



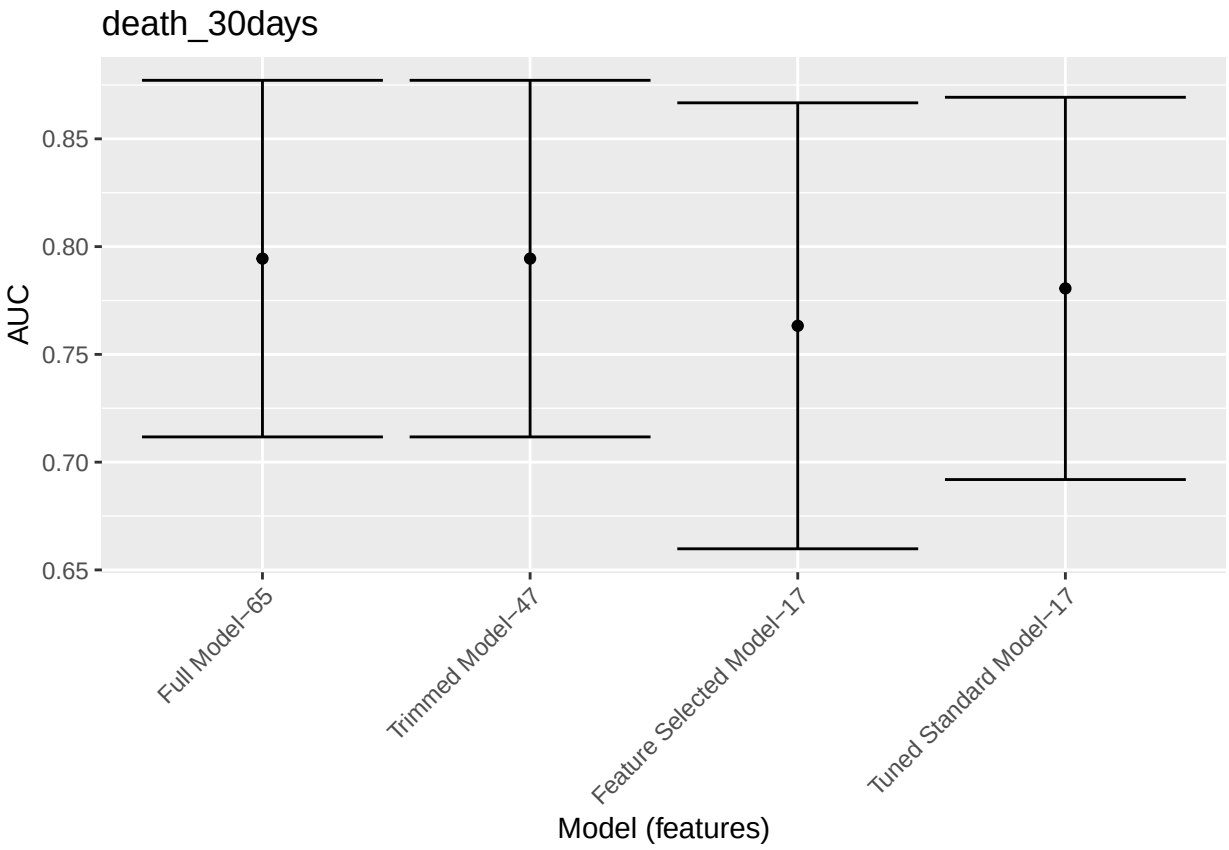

```
## $num_iterations
## [1] 284
##
## $learning_rate
```

```
## [1] 0.02775186
##
## $max_depth
## [1] 1
##
## $feature_fraction
## [1] 1
##
## $min_data_in_leaf
## [1] 21
##
## $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
##
##
## $seed
## [1] 87212
##
## $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))
```