Final Model - readmission_60d

Eduardo Yuki Yada

Global parameters

```
k <- 5 # Number of folds for cross validation
grid_size <- 30 # Number of parameter combination to tune on each model
max_auc_loss <- 0.01 # Max accepted loss of AUC for reducing num of features
```

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. age
2. education_level
3. underlying_heart_disease
4. heart_disease
5. nyha_basal
6. prior_mi
7. heart_failure
8. af
9. cardiac_arrest
10. transplant
11. valvopathy
12. diabetes

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

74. arteriografia
75. cintilografia
76. ecocardiograma
77. endoscopia
78. pet_ct
79. ultrassom
80. tomografia
81. ressonancia
82. exames_imagem_qtde
83. bic
84. 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))
```

Feature Selection

```
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")

  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)
```

```
  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: 0.706"
```

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

```
## [1] "Full Model Test AUC: 0.681"
```

Features with zero importance on the initial model:

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

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

1. education_level
2. underlying_heart_disease
3. heart_disease
4. transplant
5. hemodialysis
6. dialysis_t0
7. antiplaquetario_ev
8. insulina
9. antiviral
10. cec
11. transplante_cardiaco
12. cir_toracica
13. icp
14. angioplastia
15. cateter_venoso_central
16. teste_esforco
17. espiro_ergoespiro
18. tilt_teste
19. biopsia
20. angio_rm
21. arteriografia
22. pet_ct

```r
trimmed_features <- full_model$importance$Feature
hyperparameters$mtry <- min(hyperparameters$mtry, length(trimmed_features))
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: 0.685"
```

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

```
## [1] "Trimmed Model Test AUC: 0.663"
```

```r
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) {
  current_least_important <-
    tail(setdiff(current_model$importance$Feature, whitelist), 1)
  test_features <-
    setdiff(current_features, current_least_important)
  hyperparameters$mtry <-
    min(hyperparameters$mtry, length(test_features))
  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
  current_auc_loss <- full_model$cv_auc - current_model$cv_auc

  if (instant_auc_loss < max_auc_loss / 5 &
      current_auc_loss < max_auc_loss) {
    dropped <- TRUE
    current_features <- test_features
  } 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] "83"          "-4e-04"      "-4e-04"      "nyha_basal"
## [1] "82"               "8e-04"           "0.0012"          "education_level"
## [1] "81"        "9e-04"     "1e-04"     "angio_tc"
## [1] "80"                    "3e-04"
## [3] "-6e-04"                "cateter_venoso_central"
## [1] "79"          "-1e-04"      "-3e-04"      "transfusao"
## [1] "78"              "0.0019"          "0.0019"          "cardiac_arrest"
## [1] "77"        "0.0019"    "0"         "diabetes"
## [1] "76"                      "0.0018"
## [3] "0"                       "outros_proced_cirurgicos"
## [1] "75"         "0.0027"     "9e-04"      "citologia"
## [1] "74"            "0.0011"        "-0.0017"       "cied_final_1"
## [1] "74"        "0.0051"    "0.0041"    "digoxina"
## [1] "74"        "0.004"     "0.0029"    "cve_desf"
## [1] "74"                "0.0044"            "0.0034"
## [4] "procedure_type_1"
## [1] "74"      "0.0043" "0.0033" "icp"
## [1] "74"                "0.0048"            "0.0037"
## [4] "eletrofisiologia"
## [1] "74"          "0.0037"      "0.0026"      "tomografia"
## [1] "74"          "0.004"       "0.0029"      "valvopathy"
## [1] "74"           "0.0037"       "0.0026"       "ressonancia"
## [1] "74"           "0.0036"       "0.0025"       "antifungico"
## [1] "74"        "0.0038"    "0.0027"    "insulina"
## [1] "74"       "0.0035"   "0.0024"   "biopsia"
## [1] "74"                   "0.0046"              "0.0035"
## [4] "procedure_type_new"
## [1] "74"                    "0.0034"               "0.0023"
## [4] "ventilacao_mecanica"
## [1] "74"             "0.0031"         "0.002"          "interconsulta"
## [1] "73"        "0.0026"    "0.0016"    "prior_mi"
## [1] "72"      "0.0041" "0.0015" "holter"
## [1] "71"      "0.0053" "0.0012" "cec"
## [1] "70"           "0.0063"       "0.001"        "cateterismo"
## [1] "69"              "0.005"           "-0.0013"         "betabloqueador"
## [1] "69"                   "0.0074"               "0.0024"
## [4] "histopatologia_qtde"
## [1] "68"             "0.0067"         "0.0016"         "cintilografia"
## [1] "67"      "0.0073" "6e-04"   "bic"
## [1] "66"      "0.0086" "0.0014" "af"
## [1] "65"       "0.0093"   "7e-04"    "cultura"
## [1] "65"           "0.0128"       "0.0035"       "reop_type_1"
```

```
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	84	0.7064	0.0162	0.0000	0.0000
nyha_basal	TRUE	83	0.7068	0.0167	-0.0004	-0.0004
education_level	TRUE	82	0.7056	0.0169	0.0008	0.0012
angio_tc	TRUE	81	0.7055	0.0165	0.0009	0.0001
cateter_venoso_central	TRUE	80	0.7061	0.0165	0.0003	-0.0006
transfusao	TRUE	79	0.7064	0.0171	-0.0001	-0.0003
cardiac_arrest	TRUE	78	0.7045	0.0163	0.0019	0.0019
diabetes	TRUE	77	0.7045	0.0163	0.0019	0.0000
outros_proced_cirurgicos	TRUE	76	0.7046	0.0157	0.0018	0.0000
citologia	TRUE	75	0.7036	0.0154	0.0027	0.0009
cied_final_1	TRUE	74	0.7053	0.0159	0.0011	-0.0017
digoxina	FALSE	73	0.7012	0.0154	0.0051	0.0041
cve_desf	FALSE	73	0.7024	0.0160	0.0040	0.0029
procedure_type_1	FALSE	73	0.7019	0.0153	0.0044	0.0034
icp	FALSE	73	0.7020	0.0160	0.0043	0.0033
eletrofisiologia	FALSE	73	0.7016	0.0155	0.0048	0.0037
tomografia	FALSE	73	0.7027	0.0151	0.0037	0.0026
valvopathy	FALSE	73	0.7024	0.0157	0.0040	0.0029
ressonancia	FALSE	73	0.7027	0.0164	0.0037	0.0026
antifungico	FALSE	73	0.7028	0.0158	0.0036	0.0025
insulina	FALSE	73	0.7026	0.0155	0.0038	0.0027
biopsia	FALSE	73	0.7029	0.0158	0.0035	0.0024
procedure_type_new	FALSE	73	0.7018	0.0162	0.0046	0.0035
ventilacao_mecanica	FALSE	73	0.7030	0.0157	0.0034	0.0023
interconsulta	FALSE	73	0.7033	0.0152	0.0031	0.0020
prior_mi	TRUE	73	0.7038	0.0158	0.0026	0.0016
holter	TRUE	72	0.7023	0.0162	0.0041	0.0015
cec	TRUE	71	0.7011	0.0155	0.0053	0.0012
cateterismo	TRUE	70	0.7001	0.0158	0.0063	0.0010
betabloqueador	TRUE	69	0.7014	0.0159	0.0050	-0.0013
histopatologia_qtde	FALSE	68	0.6990	0.0164	0.0074	0.0024
cintilografia	TRUE	68	0.6997	0.0160	0.0067	0.0016
bic	TRUE	67	0.6991	0.0164	0.0073	0.0006
af	TRUE	66	0.6978	0.0157	0.0086	0.0014
cultura	TRUE	65	0.6971	0.0155	0.0093	0.0007
reop_type_1	FALSE	64	0.6936	0.0154	0.0128	0.0035

```
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)
```

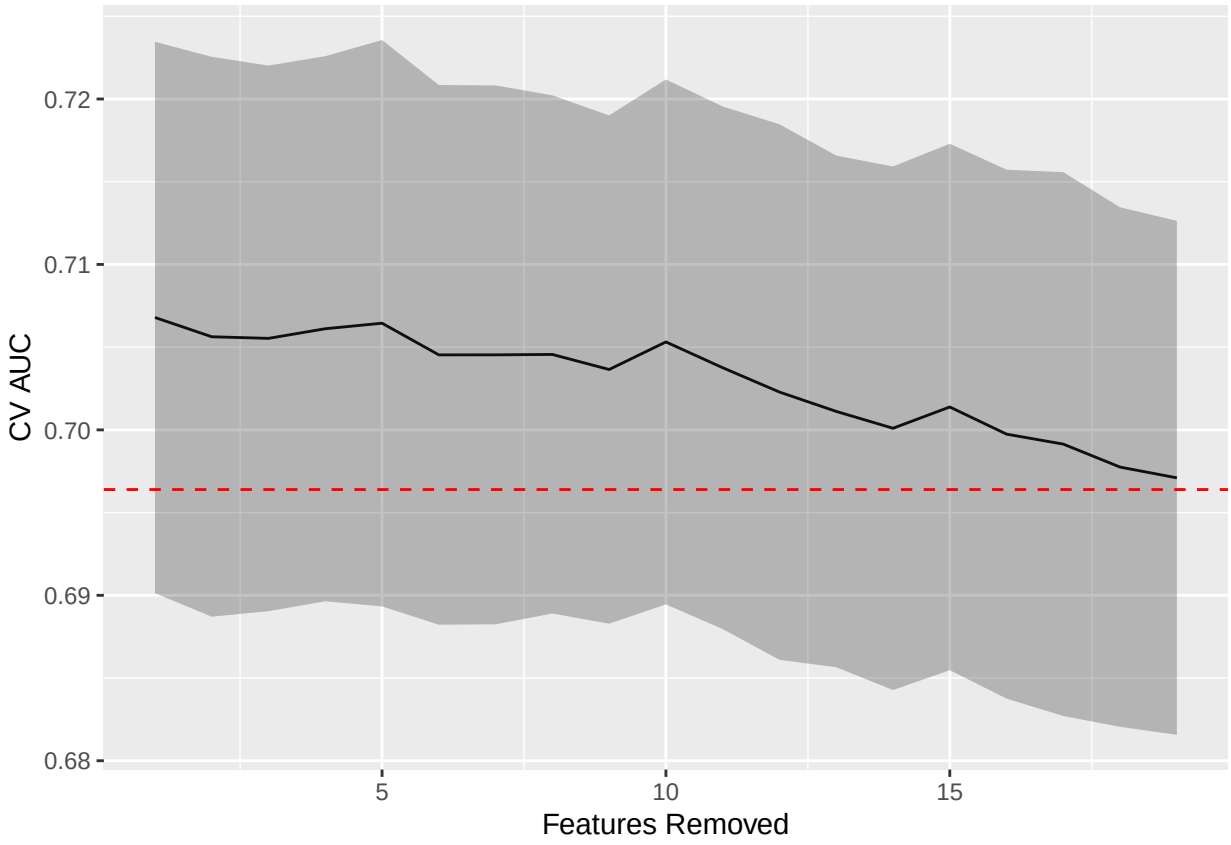
```
sprintf('Selected Model CV Train AUC: %.3f', feature_selected_model$cv_auc)
```

```
## [1] "Selected Model CV Train AUC: 0.696"
```

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

```
## [1] "Selected Model Test AUC: 0.676"
```

```
 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)
```

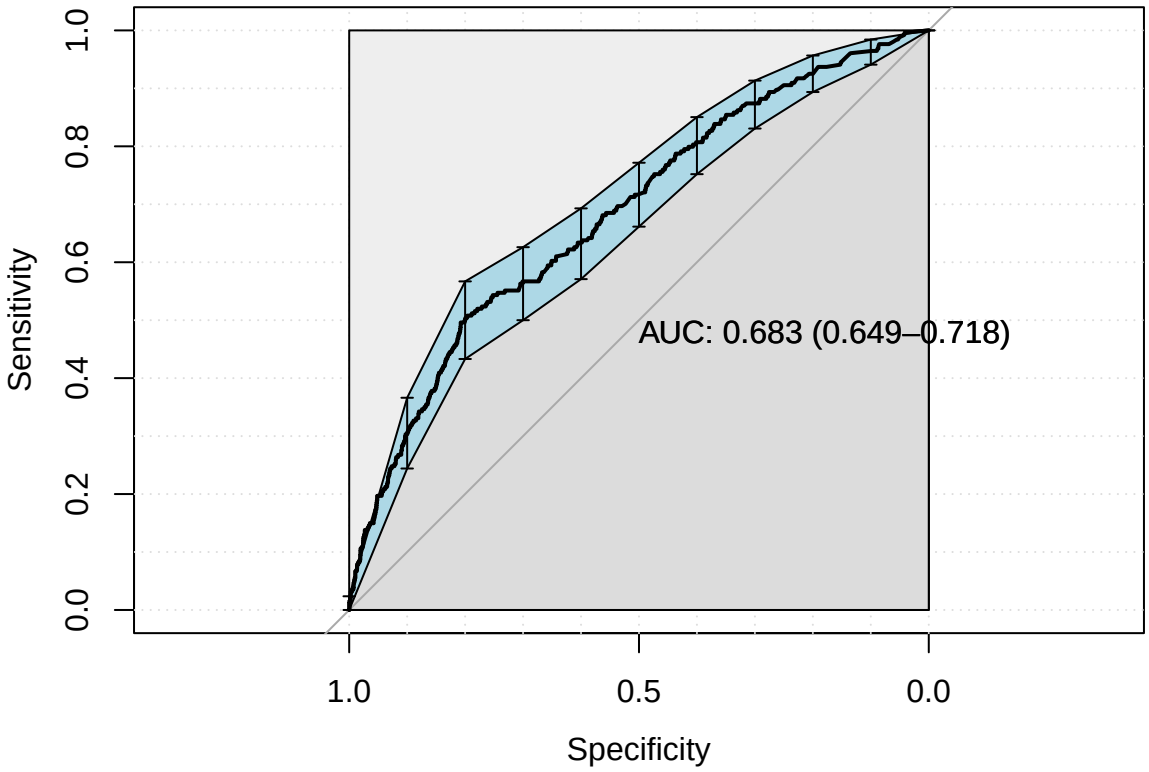
1. age
2. underlying_heart_disease
3. heart_disease
4. heart_failure
5. transplant

6. valvopathy
7. hemodialysis
8. comorbidities_count
9. procedure_type_1
10. reop_type_1
11. procedure_type_new
12. cied_final_group_1
13. admission_pre_t0_count
14. admission_pre_t0_180d
15. icu_t0
16. dialysis_t0
17. admission_t0_emergency
18. aco
19. antiarritmico
20. ieca_bra
21. dva
22. digoxina
23. estatina
24. diuretico
25. vasodilatador
26. insuf_cardiaca
27. espironolactona
28. bloq_calcio
29. antiplaquetario_ev
30. insulina
31. anticonvulsivante
32. psicofarmacos
33. antifungico
34. antiviral
35. classe_meds_qtde
36. meds_cardiovasc_qtde
37. meds_antimicrobianos
38. ventilacao_mecanica
39. transplante_cardiaco
40. cir_toracica
41. icp
42. angioplastia
43. eletrofisiologia
44. proced_invasivos_qtde
45. cve_desf
46. interconsulta
47. equipe_multiprof
48. teste_esforco
49. espiro_ergoespiro
50. tilt_teste
51. metodos_graficos_qtde
52. laboratorio
53. analises_clinicas_qtde
54. biopsia
55. histopatologia_qtde
56. angio_rm
57. arteriografia
58. ecocardiograma
59. endoscopia
60. pet_ct
61. ultrassom
62. tomografia
63. ressonancia
64. exames_imagem_qtde
65. hospital_stay

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_smote_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()) %>%
  step_impute_mean(all_numeric_predictors()) %>%
  step_smote(!!sym(outcome_column))

lightgbm_upsample_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()) %>%
  step_upsample(!!sym(outcome_column))

lightgbm_tuning <- function(recipe) {

  lightgbm_spec <- boost_tree(
    mtry = tune(),
    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(
    mtry(range = c(1L, length(selected_features))),
    trees(range = c(100L, 300L)),
    min_n(),
    tree_depth(),
    learn_rate(),
    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")

  lightgbm_tune %>%
    collect_metrics() %>%
    filter(.metric == "roc_auc") %>%
    select(mean, mtry:tree_depth) %>%
    pivot_longer(mtry:tree_depth,
                 values_to = "value",
                 names_to = "parameter"
    ) %>%
    ggplot(aes(value, mean, color = parameter)) +
    geom_point(alpha = 0.8, show.legend = FALSE) +
    facet_wrap(~parameter, scales = "free_x") +
    labs(x = NULL, y = "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, mtry, min_n, tree_depth, learn_rate, loss_reduction) %>%
    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)
```



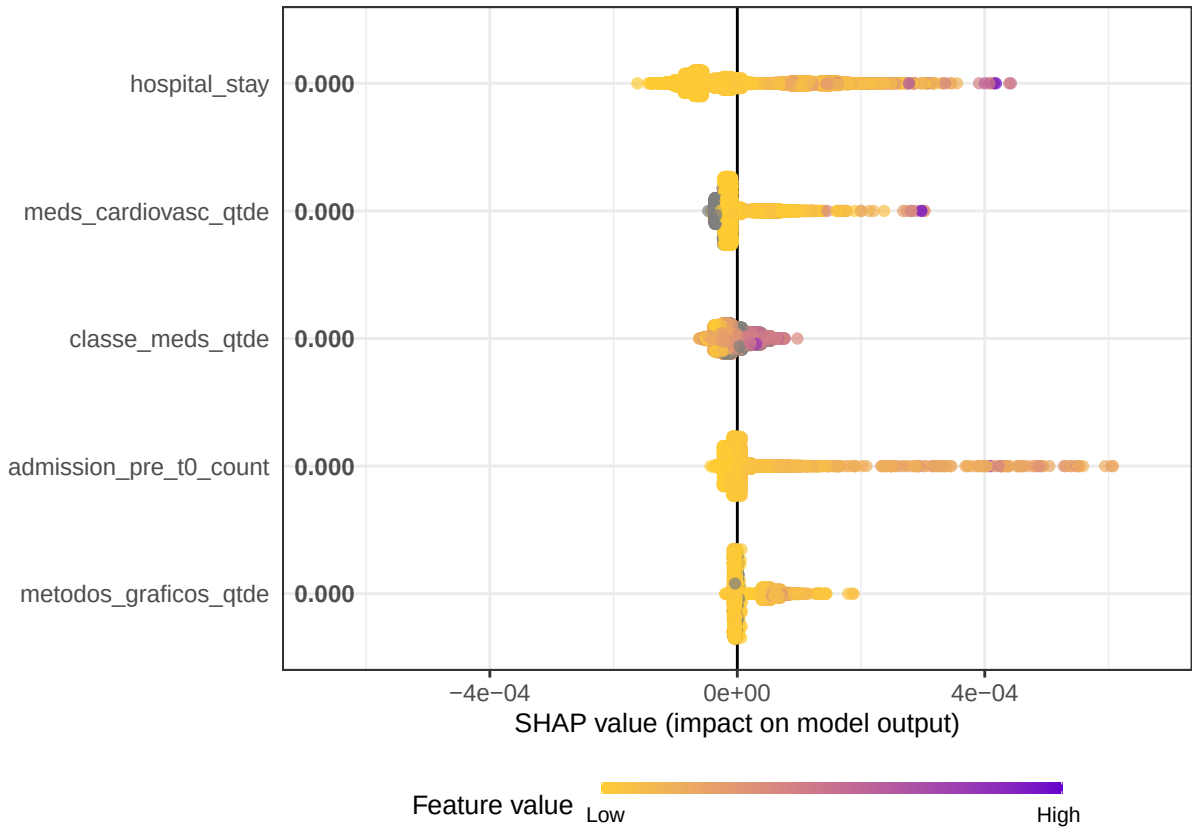

```
## [1] "Optimal Threshold: 0.06"
## Confusion Matrix and Statistics
##
##     reference
## data    0    1
##    0 3565  125
##    1  911  129
##
##                Accuracy : 0.781
##                  95% CI : (0.7689, 0.7927)
##     No Information Rate : 0.9463
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1237
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7965
##             Specificity : 0.5079
##          Pos Pred Value : 0.9661
##          Neg Pred Value : 0.1240
##              Prevalence : 0.9463
##          Detection Rate : 0.7537
##    Detection Prevalence : 0.7801
##       Balanced Accuracy : 0.6522
##
##        'Positive' Class : 0
##
```

```
# smote_results <- lightgbm_tuning(lightgbm_smote_recipe)
# upsample_results <- lightgbm_tuning(lightgbm_upsample_recipe)

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
#   )
# )
```

SHAP values

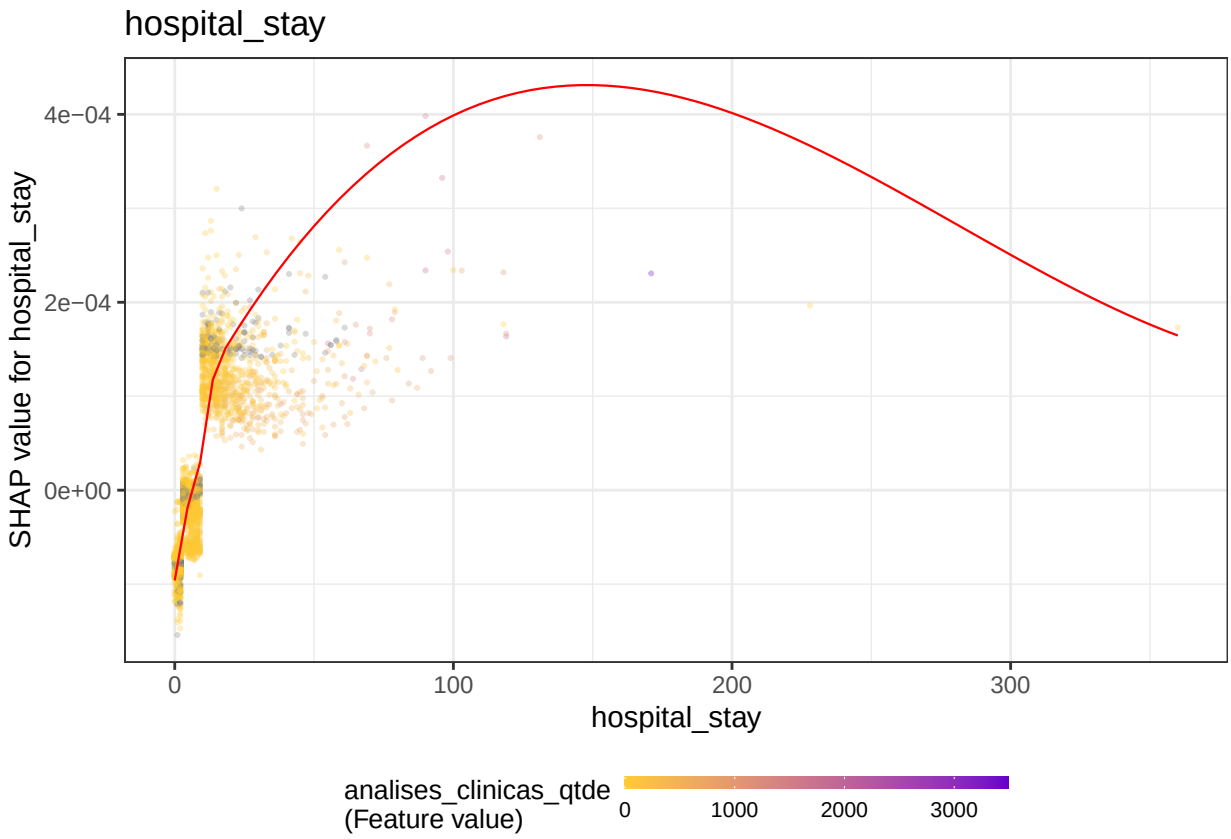
```
lightgbm_model <- parsnip::extract_fit_engine(final_lightgbm_fit)

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(5, length(selected_features))

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



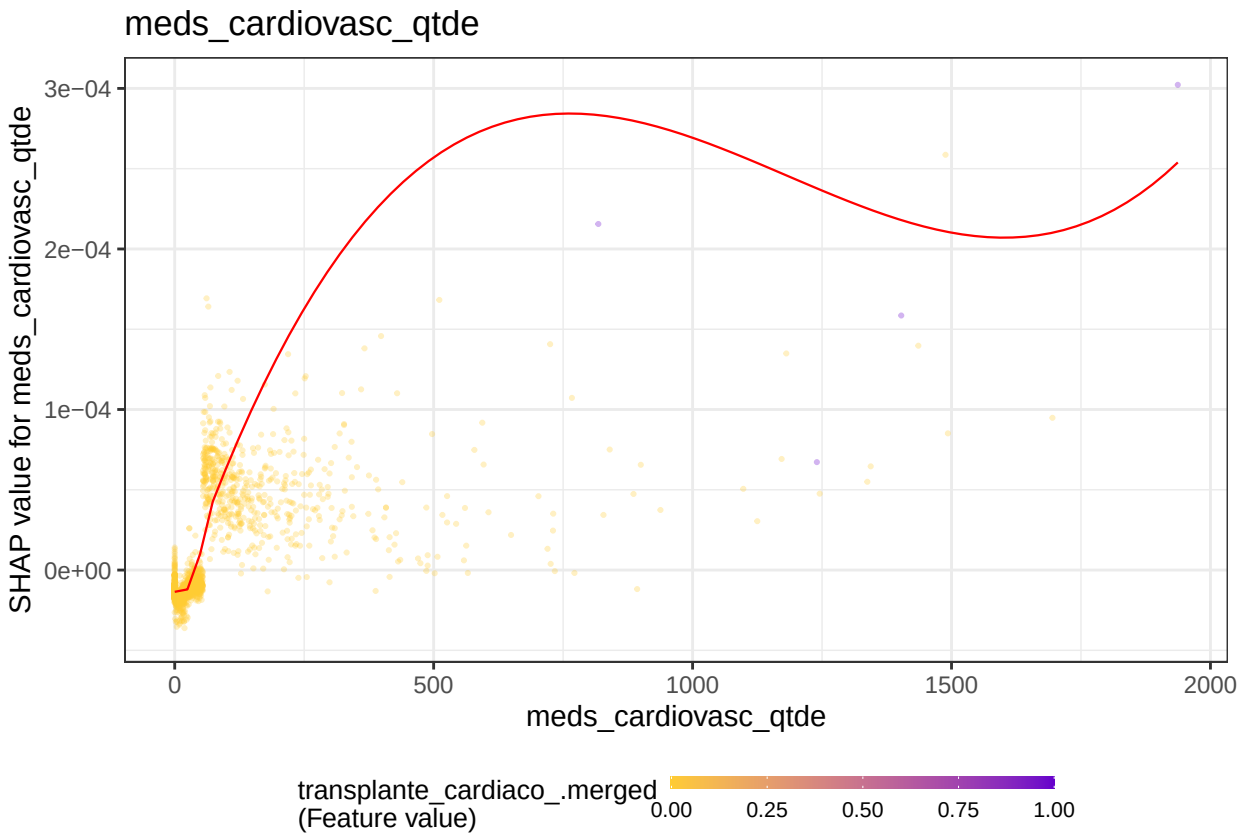

```
shap <- shap.prep(lightgbm_model, X_train = matrix_test)

for (x in shap.importance(shap, names_only = TRUE)[1:n_plots]) {
  p <- shap.plot.dependence(
    shap,
    x = x,
    color_feature = "auto",
```

```
    smooth = TRUE,
    jitter_width = 0.01,
    alpha = 0.3
  ) +
    labs(title = x)
  print(p)
}
```

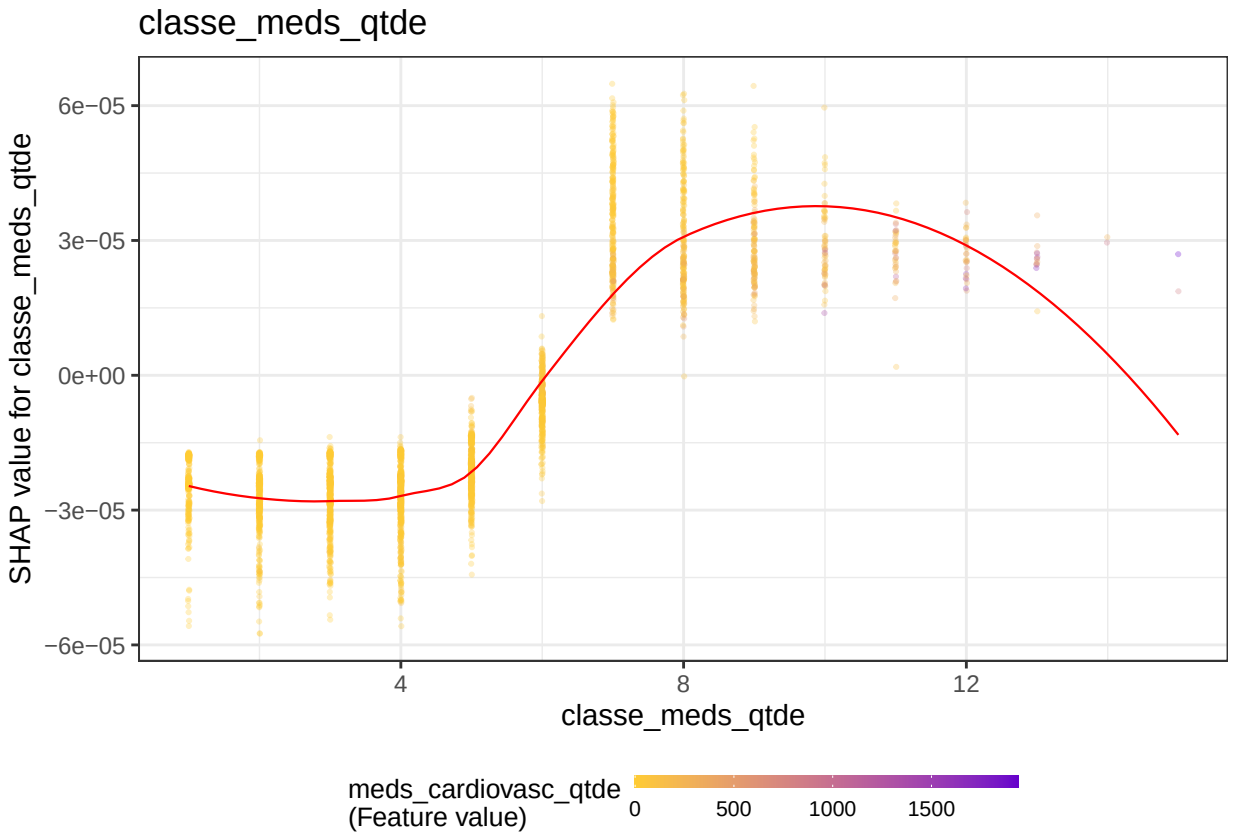
```
## `geom_smooth()` using formula 'y ~ x'
```




```
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 1041 rows containing non-finite values (stat_smooth).
## Warning: Removed 1041 rows containing missing values (geom_point).
```




```
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 1455 rows containing non-finite values (stat_smooth).
## Warning: Removed 1455 rows containing missing values (geom_point).
```



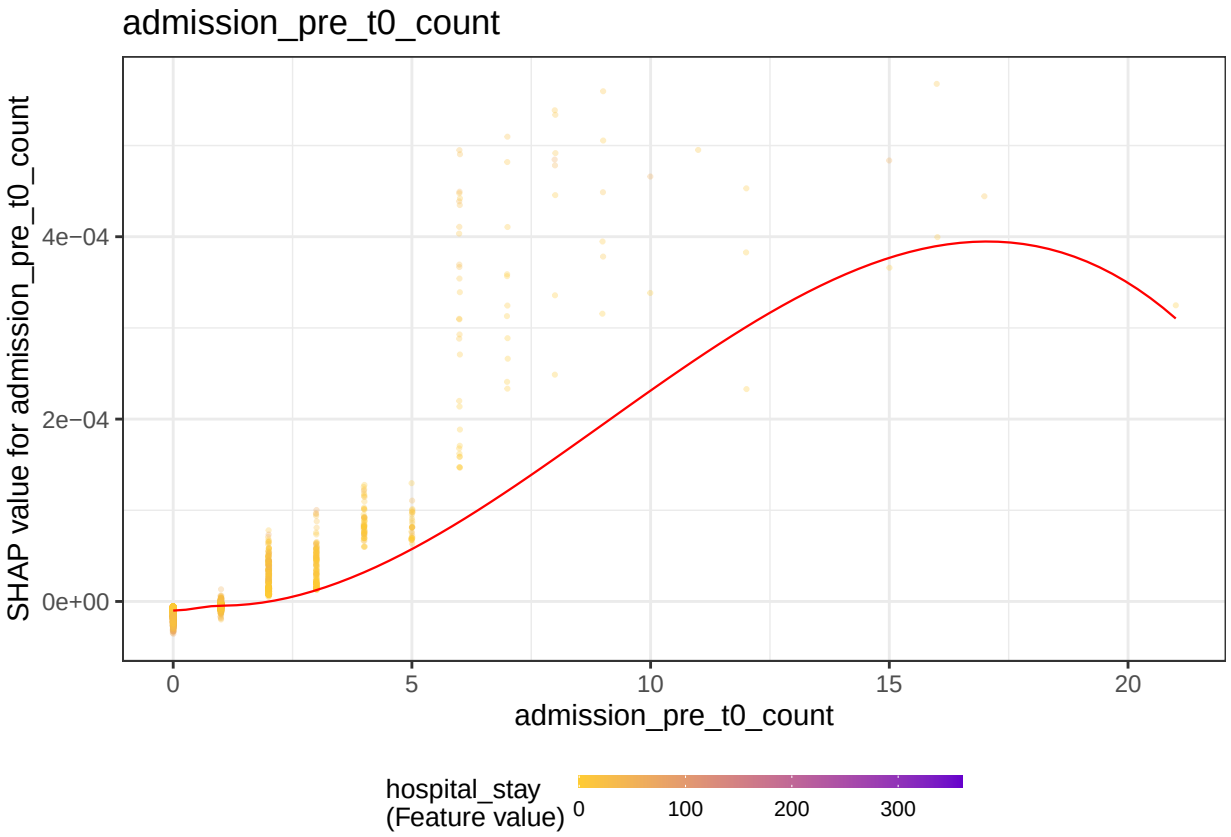

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric, : pseudoinverse used at
## -0.105
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric, : neighborhood radius
## 1.105
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric, : reciprocal condition
## number 5.2689e-29
```

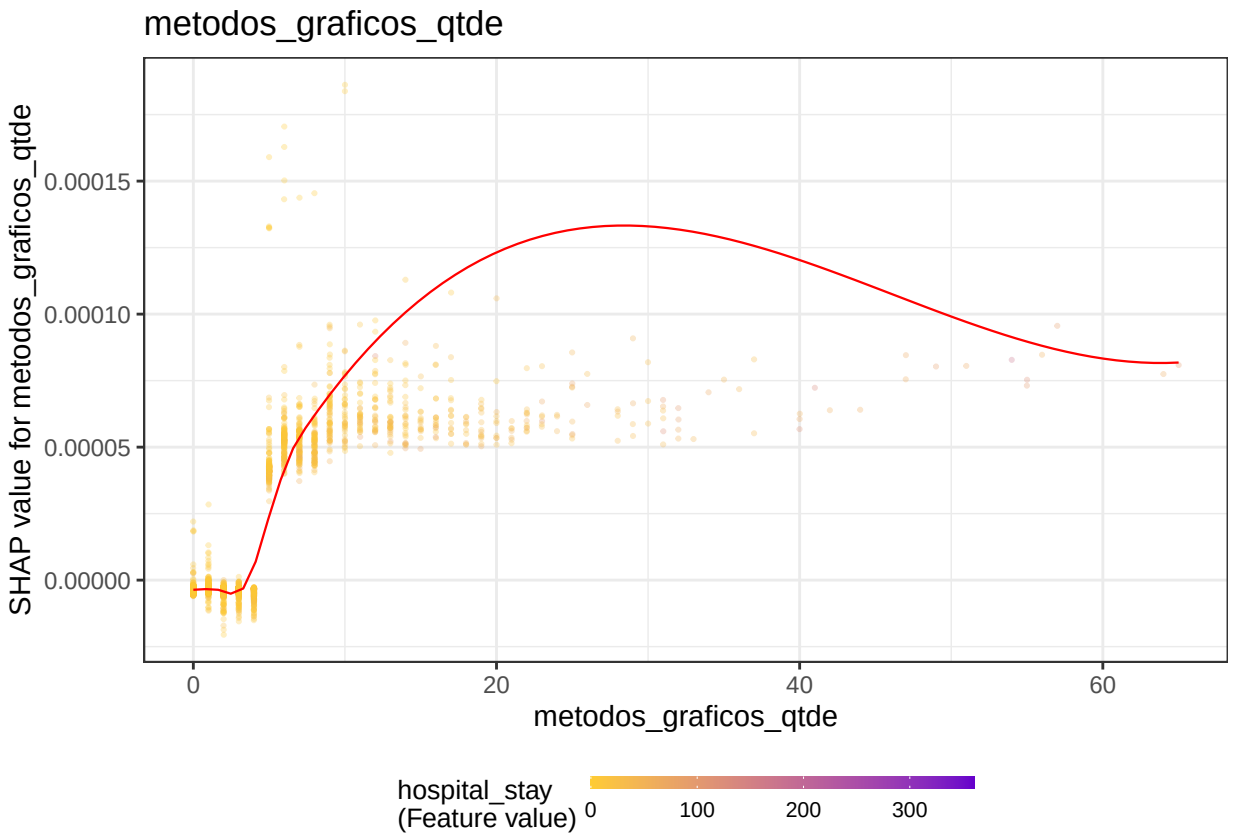
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric, : There are other near
## singularities as well. 1
```




```
## `geom_smooth()` using formula 'y ~ x'
```

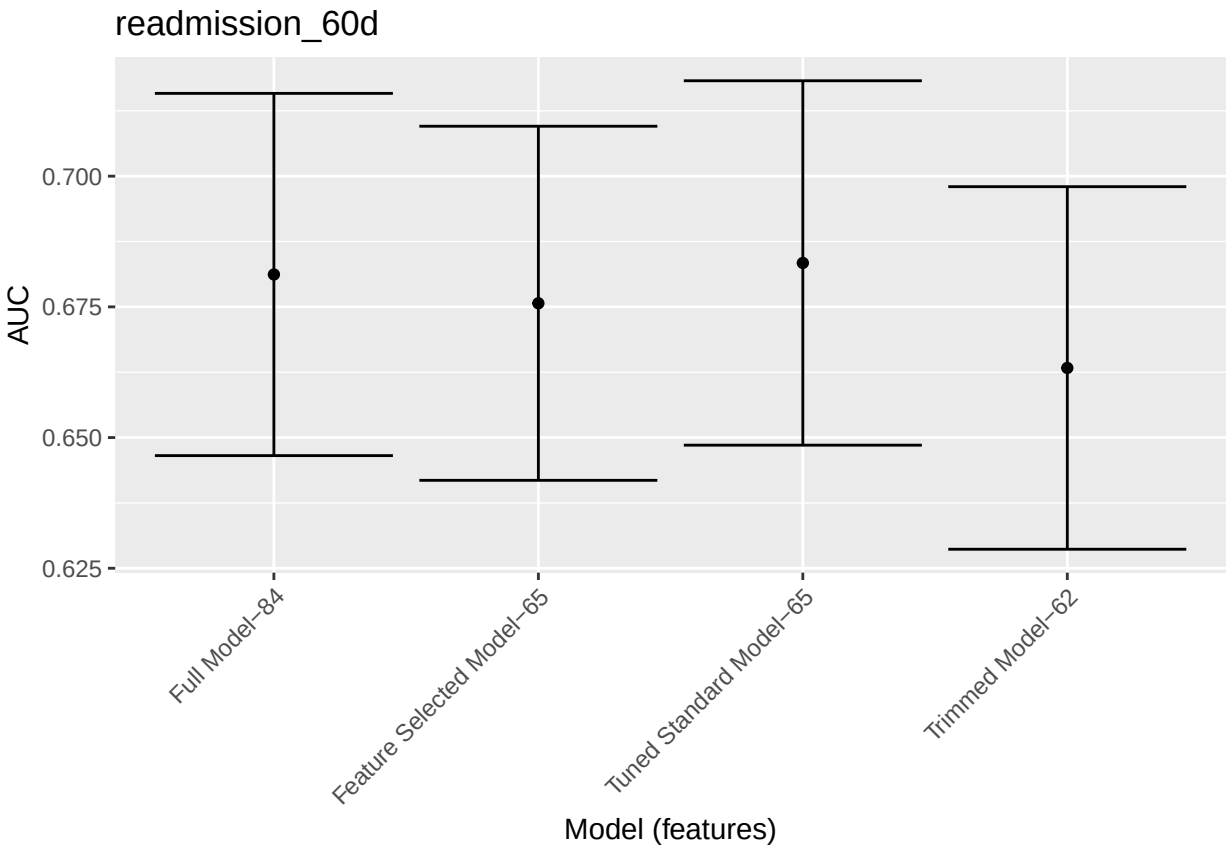
```
## Warning: Removed 802 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 802 rows containing missing values (geom_point).
```



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$auc_upper, length(s
  # 'Tuned Smote Model', smote_results$auc, smote_results$auc_lower, smote_results$auc_upper, length(selected_fe
  # 'Tuned Upsample Model', upsample_results$auc, upsample_results$auc_lower, upsample_results$auc_upper, length
) %>%
  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))
```