

Final Model - readmission_30d

Eduardo Yuki Yada

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)
```

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))
```

Eligible features

```
eligible_columns = df_names %>%
  filter(momento.aquisicao == 'Admissão t0') %>%
  .$variable.name

exception_columns = c('death_intraop', 'death_intraop_1')

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
                     )

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. 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. cec
43. transplante_cardiaco
44. outros_proced_cirurgicos
45. icp
46. intervencao_cv
47. cateterismo
48. eletrofisiologia
49. cateter Venoso_Central
50. proced_invasivos_qtde
51. cve_desf

52. transfusao
53. equipe_multiprof
54. ecg
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. radiografia
73. ressonancia
74. exames_imagem_qtde
75. bic
76. mpp

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) %>% dplyr::select(all_of(c(features, outcome_column)))
df_test <- testing(df_split) %>% dplyr::select(all_of(c(features, outcome_column)))
```

Global parameters

```
k <- 4 # Number of folds for cross validation
grid_size <- 50 # Number of parameter combination to tune on each model

set.seed(234)
df_folds <- vfold_cv(df_train, v = k,
                      strata = all_of(outcome_column))

max_auc_loss <- 0.01
```

Functions

```
niceFormatting = function(df, caption="", digits = 2, font_size = NULL){
  df %>%
    kbl(booktabs = T, longtable = T, caption = caption, digits = digits, format = "latex") %>%
    kable_styling(font_size = font_size,
                  latex_options = c("striped", "HOLD_position", "repeat_header"))
}
```

```

validation = function(model_fit, new_data, plot=TRUE) {
  library(pROC)
  library(caret)

  test_predictions_prob <-
    predict(model_fit, new_data = new_data, type = "prob") %>%
    rename_at(vars(starts_with(".pred_")), ~ str_remove(., ".pred_")) %>%
    .\$`1` 

  pROC_obj <- roc(
    new_data[[outcome_column]],
    test_predictions_prob,
    direction = "<",
    levels = c(0, 1),
    smoothed = TRUE,
    ci = TRUE,
    ci.alpha = 0.9,
    stratified = FALSE,
    plot = plot,
    auc.polygon = TRUE,
    max.auc.polygon = TRUE,
    grid = TRUE,
    print.auc = TRUE,
    show.thres = TRUE
  )
}

test_predictions_class <-
  predict(model_fit, new_data = new_data, type = "class") %>%
  rename_at(vars(starts_with(".pred_")), ~ str_remove(., ".pred_")) %>%
  .$class

conf_matrix <- table(test_predictions_class, new_data[[outcome_column]])

if (plot) {
  sens.ci <- ci.se(pROC_obj)
  plot(sens.ci, type = "shape", col = "lightblue")
  plot(sens.ci, type = "bars")

  confusionMatrix(conf_matrix) %>% print
}

return(pROC_obj)
}

```

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") %>%
    step_impute_mean(all_numeric_predictors()) %>%
    step_zv(all_predictors())

  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)

return(list(cv_auc = collect_metrics(model_fit_rs) %>% filter(.metric == 'roc_auc') %>% .$mean,
           importance = feature_importance,
           auc = as.numeric(model_auc$auc),
           auc_lower = model_auc$ci[1],
           auc_upper = model_auc$ci[3]))
}

```

```

hyperparameters <- readRDS(
  sprintf(
    "../EDA/auxiliar/model_selection/hyperparameters/lightgbm_%s.rds",
    outcome_column
  )
)

```

```
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: 0.670"
```

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

```
## [1] "Full Model Test AUC: 0.669"
```

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. nyha_basal
5. prior_mi
6. heart_failure
7. transplant
8. endocardites
9. hemodialysis
10. comorbidities_count
11. procedure_type_1
12. dialysis_t0
13. admission_t0_emergency

```
14. betabloqueador  
15. ieca_bra  
16. insuf_cardiaca  
17. espironolactona  
18. antiplaquetario_ev  
19. insulina  
20. antifungico  
21. antiviral  
22. cec  
23. transplante_cardiaco  
24. outros_proced_cirurgicos  
25. icp  
26. intervencao_cv  
27. cateterismo  
28. eletrofisiologia  
29. cateter_venoso_central  
30. proced_invasivos_qtde  
31. cve_desf  
32. transfusao  
33. holter  
34. cultura  
35. citologia  
36. biopsia  
37. angio_rm  
38. angio_tc  
39. cintilografia  
40. endoscopia  
41. flebografia  
42. pet_ct  
43. tomografia  
44. ressonancia
```

```
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: 0.653"
```

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

```
## [1] "Trimmed Model Test AUC: 0.649"
```

```
selection_results <- tibble::tribble(  
  ~`Number of Features`, ~`AUC Loss`, ~`Least Important Feature`,  
  length(features), 0, tail(full_model$importance$Feature, 1)  
)
```

```
if (full_model$cv_auc - trimmed_model$cv_auc < max_auc_loss) {  
  current_features <- trimmed_features  
  current_model <- trimmed_model  
  current_least_important <- tail(current_model$importance$Feature, 1)  
  current_auc_loss <- full_model$cv_auc - current_model$cv_auc  
  
  selection_results <- selection_results %>%  
    add_row(`Number of Features` = length(trimmed_features),  
           `AUC Loss` = current_auc_loss,  
           `Least Important Feature` = current_least_important)  
} else {  
  current_features <- features  
  current_model <- full_model
```

```

current_least_important <- tail(current_model$importance$Feature, 1)
current_auc_loss <- 0
}

while (current_auc_loss < max_auc_loss) {
  last_feature_dropped <- current_least_important

  current_features <- setdiff(current_features, current_least_important)
  hyperparameters$mtry = min(hyperparameters$mtry, length(current_features))
  current_model <- model_fit_wf(df_train, current_features, outcome_column, hyperparameters)
  current_least_important <- tail(current_model$importance$Feature, 1)

  current_auc_loss <- full_model$cv_auc - current_model$cv_auc

  selection_results <- selection_results %>%
    add_row(`Number of Features` = length(current_features),
           `AUC Loss` = current_auc_loss,
           `Least Important Feature` = current_least_important)

  # print(c(length(current_features), current_auc_loss))
}

selection_results %>% niceFormatting(digits = 4)

```

Table 1:

Number of Features	AUC Loss	Least Important Feature
76	0.0000	mpp
75	-0.0031	procedure_type_1
74	-0.0030	proced_invasivos_qtde
73	-0.0039	digoxina
72	-0.0052	procedure_type_new
71	-0.0014	ecocardiograma
70	-0.0002	biopsia
69	-0.0020	ultrassom
68	-0.0031	antiviral
67	-0.0025	estatina
66	0.0005	angio_tc
65	0.0013	insuf_cardiaca
64	-0.0072	cied_final_group_1
63	-0.0063	reop_type_1
62	0.0047	cve_desf
61	0.0076	comorbidities_count
60	0.0040	antifungico
59	0.0002	espironolactona
58	0.0030	cultura
57	0.0058	bloq_calcio
56	0.0032	dva
55	0.0054	antiarritmico
54	0.0002	aco
53	0.0018	histopatologia_qtde
52	0.0038	angio_rm
51	0.0068	ieca_bra
50	0.0010	radiografia
49	0.0024	admission_pre_t0_180d
48	-0.0010	classe_meds_qtde
47	0.0129	cied_final_1

```

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

con <- file(sprintf('../EDA/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)

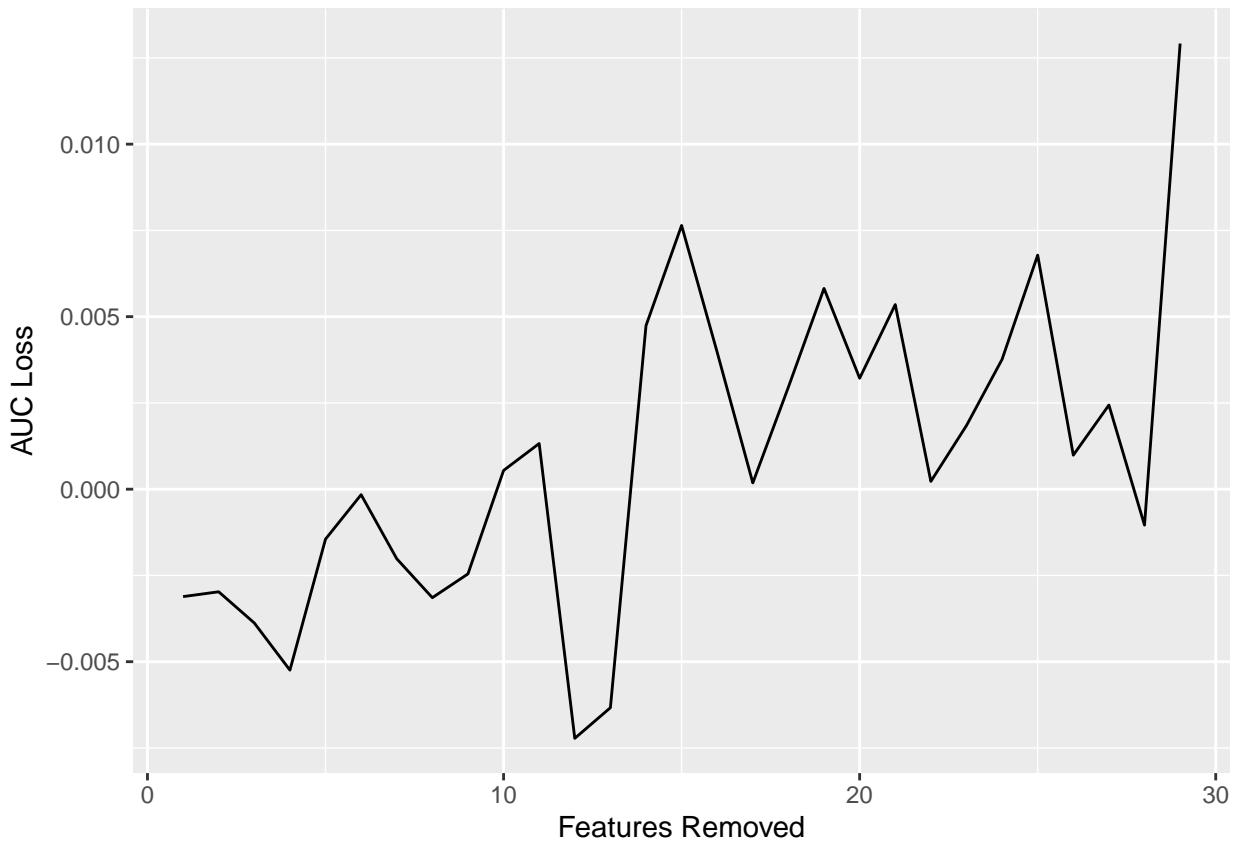
sprintf('Trimmed Model CV Train AUC: %.3f', feature_selected_model$cv_auc)

## [1] "Trimmed Model CV Train AUC: 0.659"
sprintf('Trimmed Model Test AUC: %.3f', feature_selected_model$auc)

## [1] "Trimmed Model Test AUC: 0.654"

selection_results %>%
  filter(`Number of Features` < length(features)) %>%
  mutate(`Features Removed` = length(features) - `Number of Features`) %>%
  ggplot(aes(x = `Features Removed`, y = `AUC Loss`)) +
  geom_line()

```



Hyperparameter tuning

Selected features:

```
gluedown::md_order(selected_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. cied_final_1
11. admission_pre_t0_count
12. icu_t0
13. dialysis_t0
14. admission_t0_emergency
15. betabloqueador
16. diuretico
17. vasodilatador
18. antiplaquetario_ev
19. insulina
20. anticonvulsivante
21. psicofarmacos
22. meds_cardiovasc_qtd
23. meds_antimicrobianos
24. cec
25. transplante_cardiaco
26. outros_proced_cirurgicos
27. icp
28. intervencao_cv
29. cateterismo
30. eletrofisiologia
31. cateter_venoso_central
32. transfusao
33. equipe_multiprof
34. ecg
35. holter
36. metodos_graficos_qtd
37. laboratorio
38. analises_clinicas_qtd
39. citologia
40. cintilografia
41. endoscopia
42. flebografia
43. pet_ct
44. tomografia
45. ressonancia
46. exames_imagem_qtd
47. bic

```

```

lightgbm_recipe <-
  recipe(formula = sprintf("%s ~ .", outcome_column) %>% as.formula,
         data = df_train %>% dplyr::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(), one_hot = TRUE) %>%
  step_impute_mean(all_numeric_predictors()) %>%
  step_zv(all_predictors())

lightgbm_spec <- boost_tree(
  mtry = tune(),
  trees = tune(),
  min_n = tune(),
  tree_depth = tune(),

```

```

learn_rate = tune(),
loss_reduction = tune()
) %>%
set_engine("lightgbm") %>%
set_mode("classification")

lightgbm_grid <- grid_latin_hypercube(
finalize(mtry(),
df_train %>% dplyr::select(all_of(c(selected_features, outcome_column)))),
dials::trees(range = c(100L, 300L)),
min_n(),
tree_depth(),
learn_rate(),
loss_reduction(),
size = grid_size
)

lightgbm_workflow <-
workflow() %>%
add_recipe(lightgbm_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)

```

Table 2:

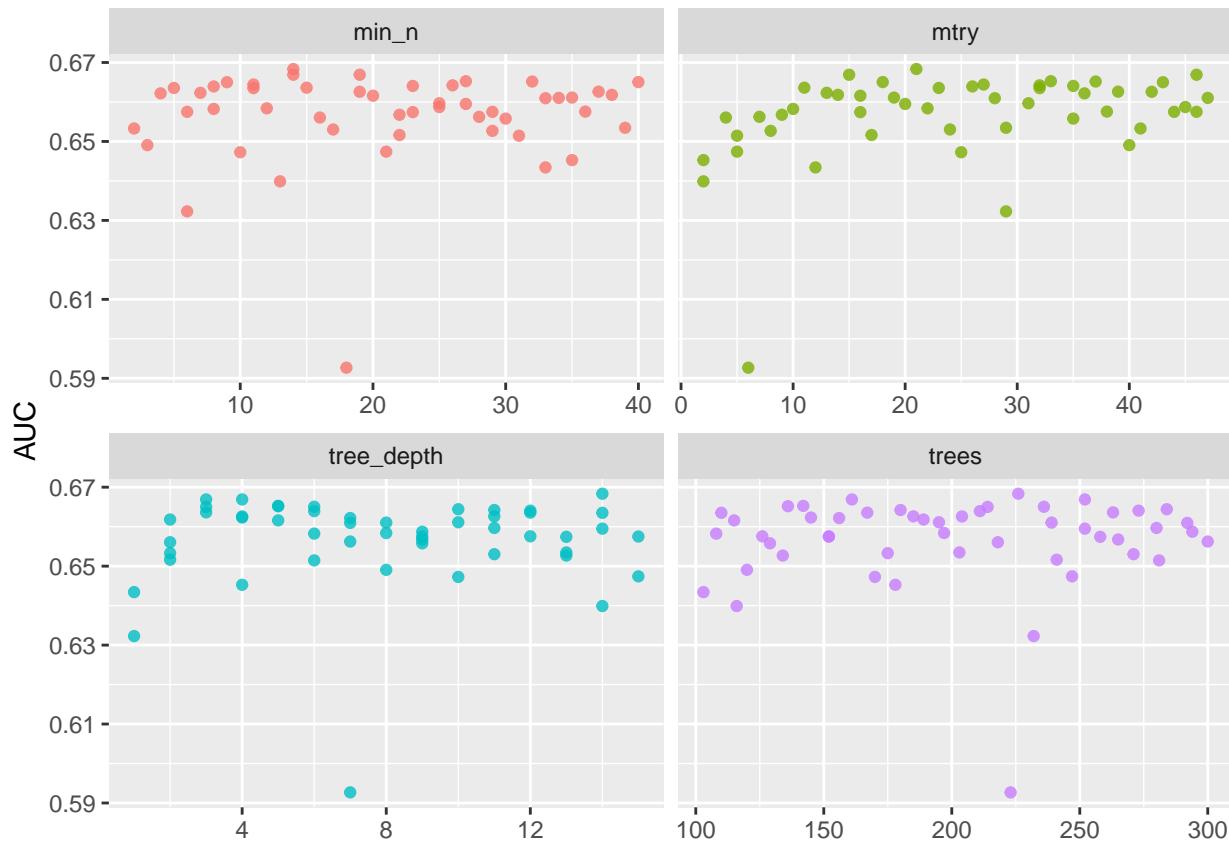
mtry	trees	min_n	tree_depth	learn_rate	loss_reduction	.metric	.estimator	mean	n	std_err	.config
21	226	14	14	0.00010	16.18911	roc_auc	binary	0.66835	4	0.00555	Preprocessor1
46	252	19	4	0.00004	0.00000	roc_auc	binary	0.66690	4	0.00644	Preprocessor1
15	161	14	3	0.01302	0.00150	roc_auc	binary	0.66690	4	0.00573	Preprocessor1
33	142	27	5	0.00000	0.00501	roc_auc	binary	0.66526	4	0.00543	Preprocessor1
37	136	32	5	0.00002	0.00000	roc_auc	binary	0.66518	4	0.00728	Preprocessor1

```

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")

```



```

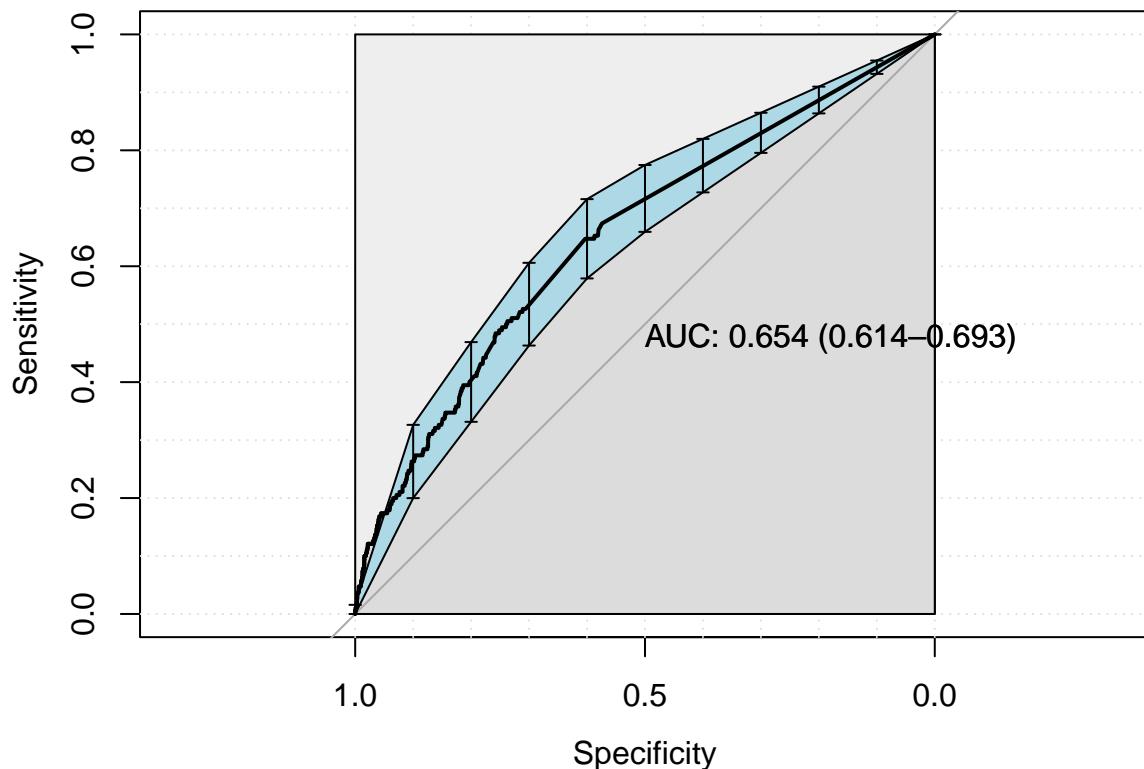
final_lightgbm_workflow <-
  lightgbm_workflow %>%
  finalize_workflow(best_lightgbm)

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)

```



```

## Confusion Matrix and Statistics
##
## test_predictions_class      0      1
##                      0 4540  190
##                      1     0     0
##
##          Accuracy : 0.9598
## 95% CI : (0.9538, 0.9652)
##  No Information Rate : 0.9598
## P-Value [Acc > NIR] : 0.5193
##
##          Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##          Sensitivity : 1.0000
##          Specificity  : 0.0000
##  Pos Pred Value  : 0.9598
##  Neg Pred Value  :    NaN
##          Prevalence  : 0.9598
##  Detection Rate  : 0.9598
## Detection Prevalence : 1.0000
##  Balanced Accuracy : 0.5000
##
##  'Positive' Class : 0
##

lightgbm_parameters <- lightgbm_tune %>%
  show_best("roc_auc", n = 1) %>%
  select(trees, mtry, min_n, tree_depth, learn_rate, loss_reduction) %>%
  as.list

saveRDS(

```

```

    lightgbm_parameters,
    file = sprintf(
      "../EDA/auxiliar/final_model/hyperparameters/lightgbm_%s.rds",
      outcome_column
    )
)

```

SHAP values

```

lightgbm_model <- parsnip::extract_fit_engine(final_lightgbm_fit)

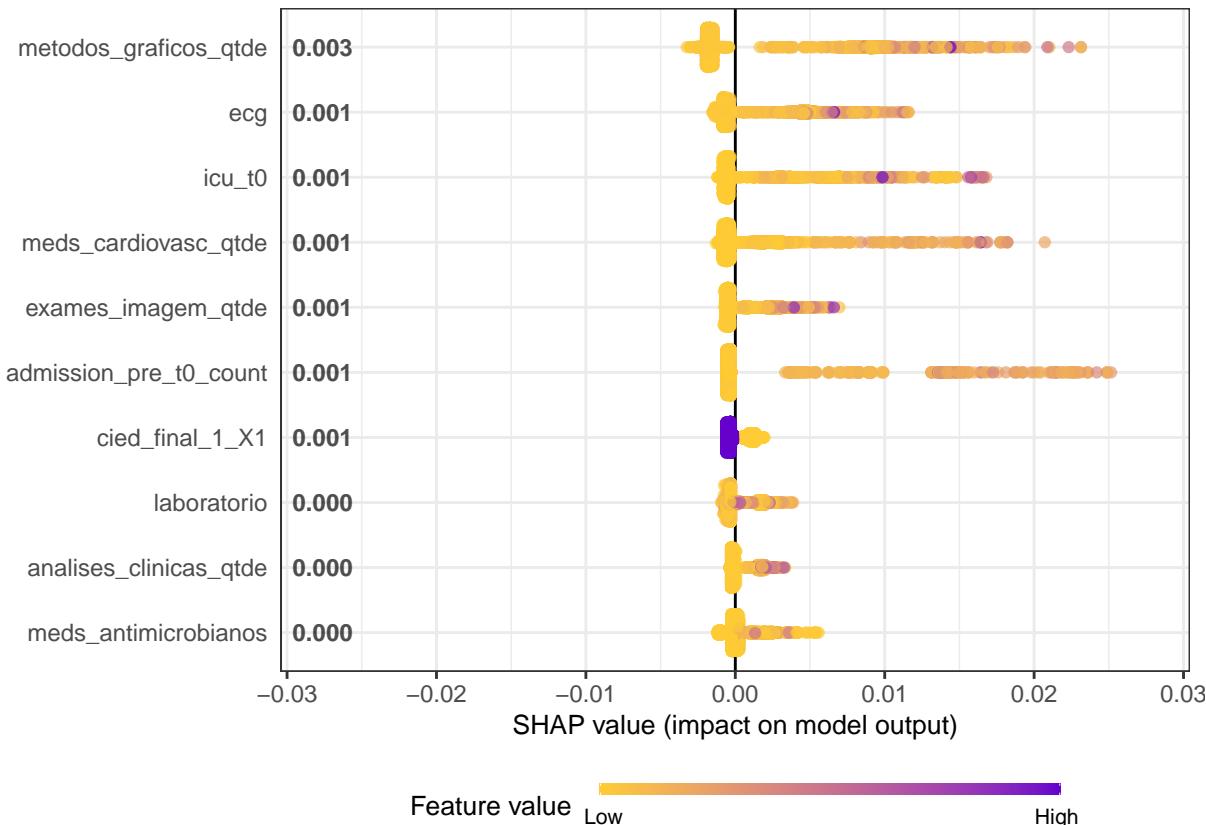
trained_rec <- prep(lightgbm_recipe, training = df_train)

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)))

shap.plot.summary.wrap1(model = lightgbm_model, X = matrix_train, top_n = 10, dilute = F)

```



```

# Crunch SHAP values
shap <- shap.prep(lightgbm_model, X_train = matrix_test)

for (x in shap.importance(shap, names_only = TRUE)[1:5]) {
  p <- shap.plot.dependence(
    shap,
    x = x,
    color_feature = "auto",
    smooth = TRUE,
    jitter_width = 0.01,
    alpha = 0.3
  )
}

```

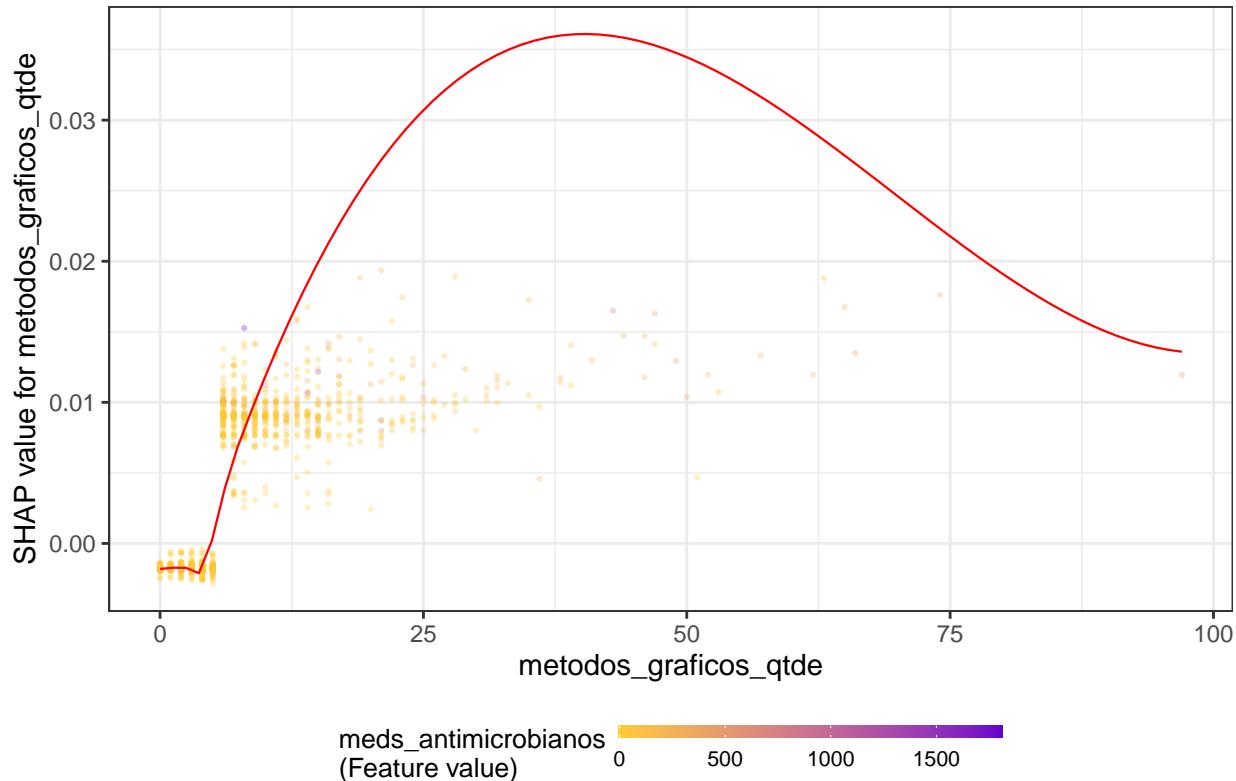
```

) +
  labs(title = x)
print(p)
}

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

```

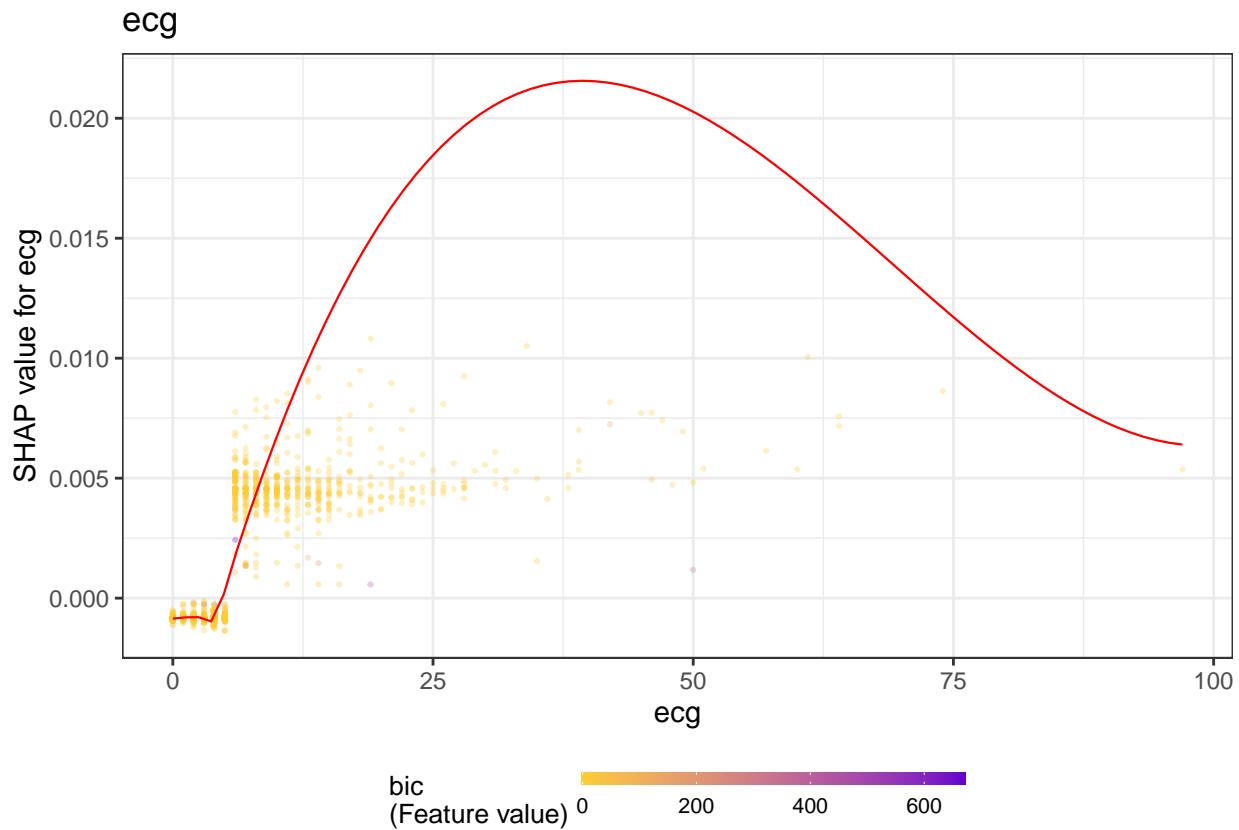
metodos_graficos_qtde



```

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

```



```

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

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : at -0.45

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : radius 0.2025

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at -0.45

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.45

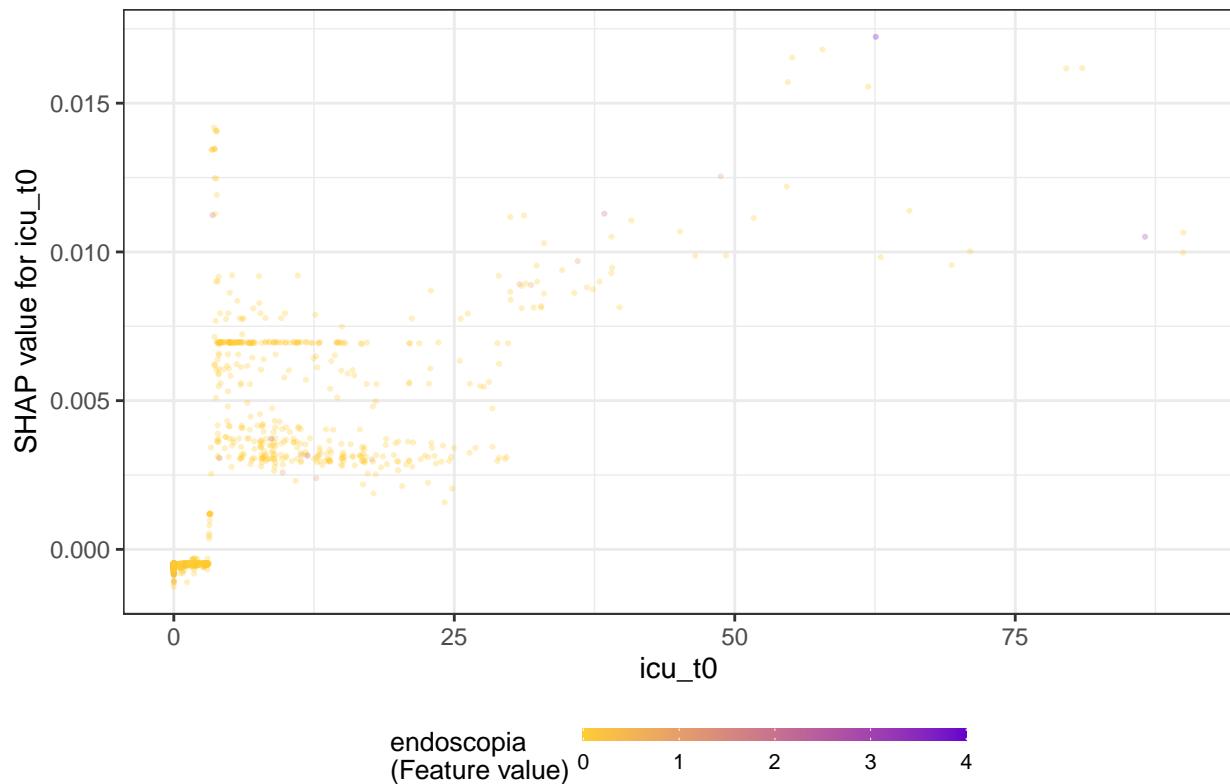
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : zero-width neighborhood. make span bigger

## Warning: Computation failed in 'stat_smooth()':
## NA/NaN/Inf in foreign function call (arg 5)

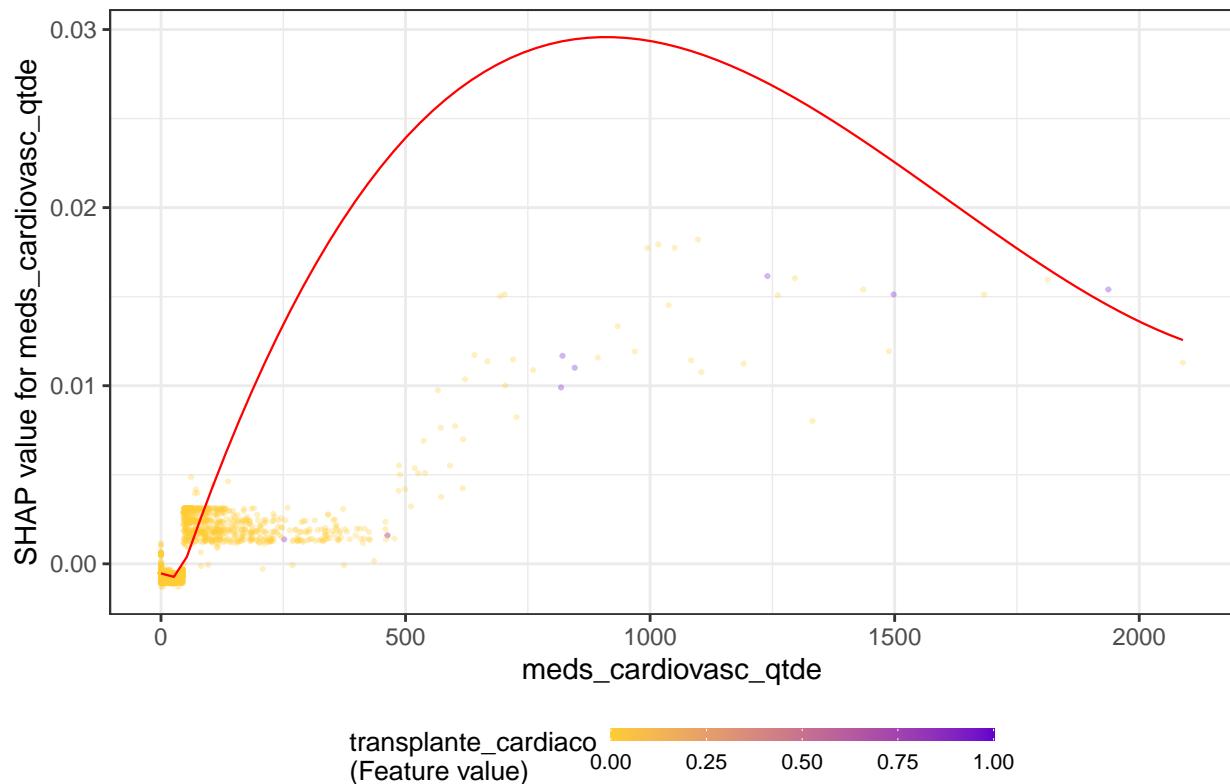
```

icu_t0

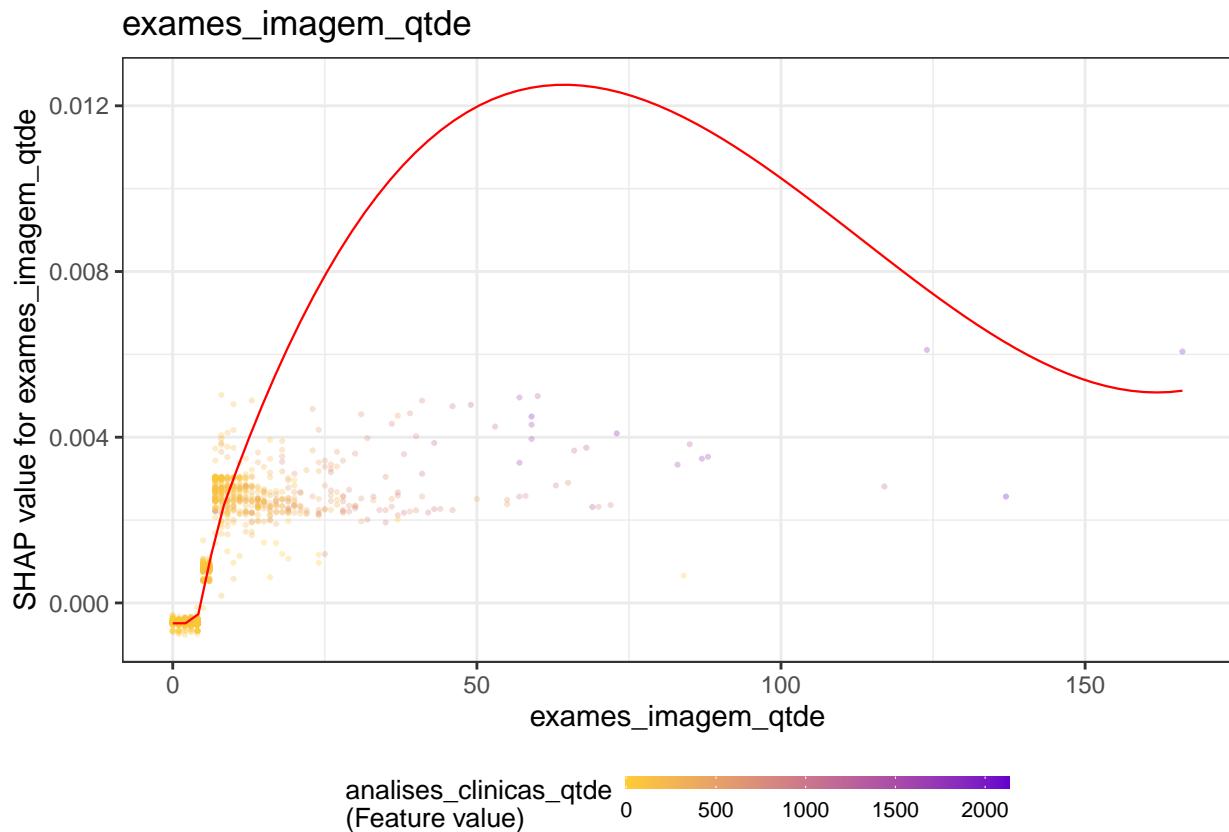


```
## `geom_smooth()` using formula 'y ~ x'
```

meds_cardiovasc_qtde



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

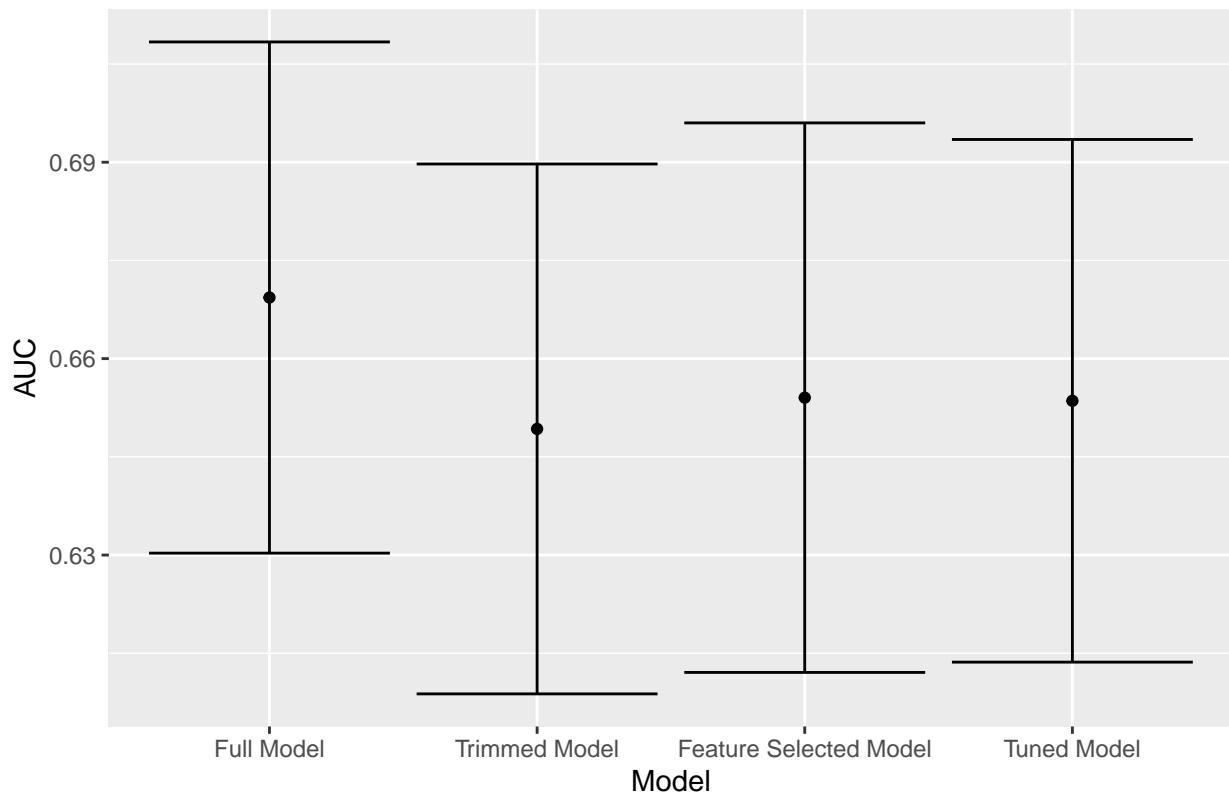


Models Comparison

```
df_auc <- tibble::tribble(
  ~Model, ~`AUC`, ~`Lower Limit`, ~`Upper Limit`,
  'Full Model', full_model$auc, full_model$auc_lower, full_model$auc_upper,
  'Trimmed Model', trimmed_model$auc, trimmed_model$auc_lower, trimmed_model$auc_upper,
  'Feature Selected Model', feature_selected_model$auc, feature_selected_model$auc_lower, feature_selected_model$auc_upper,
  'Tuned Model', as.numeric(lightgbm_auc$auc), lightgbm_auc$ci[1], lightgbm_auc$ci[3],
  # 'Oversampled Model', as.numeric(oversampled_model$auc), oversampled_model$auc_lower, oversampled_model$auc_upper,
  # 'Undersampled Model', as.numeric(undersampled_model$auc), undersampled_model$auc_lower, undersampled_model$auc_upper
) %>%
  mutate(Target = outcome_column,
    Model = factor(Model,
      levels = c('Full Model', 'Trimmed Model',
      'Feature Selected Model', 'Tuned Model',
      'Oversampled Model', 'Undersampled Model')))

df_auc %>%
  ggplot(aes(
    x = Model,
    y = AUC,
    ymin = `Lower Limit`,
    ymax = `Upper Limit`
  )) +
  geom_point() +
  geom_errorbar() +
  labs(title = outcome_column)
```

readmission_30d



```
saveRDS(df_auc, sprintf("../EDA/auxiliar/final_model/performance/%s.RData", outcome_column))
```