R 프로그래밍 기초 XGBoost v1.1

한상곤(sangkon@pusan.ac.kr)

2023.06.30(화)

Contents

1 Palmer Penguins과 Tidymodels을 활용한 분류

이 프로젝트를 진행하기 위해선 아래 주요 패키지가 필요합니다. 해당 패키지를 먼저 설치하시고 아래 문서를 진행해주세요. 이 중에서 특히 `tidyverse`, `tidymodels`을 주로 활용할 예정이며, 머신러닝 알고리즘으로 `xgboost`를 사용합니다. `palmerpenguins` 데이터에 적용하도록 하겠습니다.

```
install.packages("tidyverse")
install.packages("tidymodels")
install.packages("tictoc")
install.packages("doParallel")
install.packages("furrr")
install.packages("vip")
install.packages("finetune")
install.packages("ranger")
install.packages("glmnet")
install.packages("xgboost")
install.packages("palmerpenguins")
install.packages("hrbrthemes")
```

1.1 패키지 불러오기

hrbrthemes를 사용할 경우 roboto 폰트를 필요로 합니다. hrbrthemes::import_roboto_condensed()를 실행해서 해당 폰트를 설치해주시면 됩니다. 자세한 사항은 https://github.com/hrbrmstr/hrbrthemes나 https://cinc.rud.is/web/packages/hrbrthemes/를 참고하세요.

```
library(tidyverse)
library(tidymodels)
library(tictoc)
library(doParallel)
library(finetune)
library(furrr)
library(vip)
library(xgboost)
library(ranger)
library(glmnet)
library(hrbrthemes)
hrbrthemes::import_roboto_condensed()
```

1.2 테마 설정 및 계산 속도 향상을 위한 설정

R에서 사용하는 대부분의 알고리즘은 GPU 가속을 자주 활용하지 않기 때문에 가능하다면 CPU의 모든 자원을 활용할 수 있도록 환경을 정의하도록 하겠습니다.

```
theme_set(hrbrthemes::theme_ipsum_rc())
plan(multicore, workers = availableCores())
#plan(multiprocess, workers = availableCores())
cores <- parallel::detectCores(logical = FALSE)
cl <- makePSOCKcluster(cores)
registerDoParallel(cores = cl)
set.seed(42)
```

1.3 Data Wrangling

1.3.1 데이터 불러오기

```
penguins_data <- palmerpenguins::penguins
```

1.3.2 NA값 확인

```
glimpse(penguins_data)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adel~
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgerse~
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1, ~
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1, ~
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186~
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, ~
## $ sex               <fct> male, female, female, NA, female, male, female, male~
## $ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007~
```

```
t(map_df(penguins_data, ~sum(is.na(.))))
```

```
##                   [,1]
```

```
## species              0
## island               0
## bill_length_mm       2
## bill_depth_mm        2
## flipper_length_mm    2
## body_mass_g          2
## sex                 11
## year                 0
```

1.3.3 데이터 전처리

```
penguins_df <-
  penguins_data %>%
  filter(!is.na(sex)) %>%
  select(-year, -island)
head(penguins_df)
```

```
## # A tibble: 6 x 6
##   species bill_length_mm bill_depth_mm flipper_length_mm body_mass_g sex
##   <fct>            <dbl>         <dbl>             <int>       <int> <fct>
## 1 Adelie            39.1          18.7               181        3750 male
## 2 Adelie            39.5          17.4               186        3800 female
## 3 Adelie            40.3          18                 195        3250 female
## 4 Adelie            36.7          19.3               193        3450 female
## 5 Adelie            39.3          20.6               190        3650 male
## 6 Adelie            38.9          17.8               181        3625 female
```

```
glimpse(penguins_df)
```

```
## Rows: 333
## Columns: 6
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adel~
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, 36.7, 39.3, 38.9, 39.2, 41.1, 38.6~
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, 19.3, 20.6, 17.8, 19.6, 17.6, 21.2~
## $ flipper_length_mm <int> 181, 186, 195, 193, 190, 181, 195, 182, 191, 198, 18~
## $ body_mass_g       <int> 3750, 3800, 3250, 3450, 3650, 3625, 4675, 3200, 3800~
## $ sex               <fct> male, female, female, female, male, female, male, fe~
```

```
t(map_df(penguins_df, ~sum(is.na(.))))
```

```
##                   [,1]
## species              0
## bill_length_mm       0
## bill_depth_mm        0
## flipper_length_mm    0
## body_mass_g          0
## sex                  0
```

1.3.4 학습 데이터 및 검증 데이터

```
penguins_split <-
  rsample::initial_split(
    penguins_df,
    prop = 0.7,
    strata = species
```

```
  )
```

1.4 기준(BaselineExperiment) 설정

```
tic("1. Baseline XGBoost training duration")
xgboost_fit <-
  boost_tree() %>%
  set_engine("xgboost") %>%
  set_mode("classification") %>%
  fit(species ~ ., data = training(penguins_split))
toc(log = TRUE)
```

```
## 1. Baseline XGBoost training duration: 0.03 sec elapsed
```

```
preds <- predict(xgboost_fit, new_data = testing(penguins_split))
actual <- testing(penguins_split) %>% select(species)
yardstick::f_meas_vec(truth = actual$species, estimate = preds$.pred_class)
```

```
## [1] 1
```

1.5 모델 설정

```
ranger_model <-
  parsnip::rand_forest(mtry = tune(), min_n = tune()) %>%
  set_engine("ranger") %>%
  set_mode("classification")

glm_model <-
  parsnip::multinom_reg(penalty = tune(), mixture = tune()) %>%
  set_engine("glmnet") %>%
  set_mode("classification")

xgboost_model <-
  parsnip::boost_tree(mtry = tune(), learn_rate = tune()) %>%
  set_engine("xgboost") %>%
  set_mode("classification")

hardhat::extract_parameter_dials(glm_model, "mixture")
```

```
## Proportion of Lasso Penalty (quantitative)
## Range: [0, 1]
```

1.6 Grid 검색

```
ranger_grid <-
  hardhat::extract_parameter_set_dials(ranger_model) %>%
  finalize(select(training(penguins_split), -species)) %>%
  grid_regular(levels = 4)
ranger_grid
```

```
## # A tibble: 16 x 2
##     mtry min_n
##    <int> <int>
##  1     1     2
```

```
##  2     2     2
##  3     3     2
##  4     5     2
##  5     1    14
##  6     2    14
##  7     3    14
##  8     5    14
##  9     1    27
## 10     2    27
## 11     3    27
## 12     5    27
## 13     1    40
## 14     2    40
## 15     3    40
## 16     5    40
```

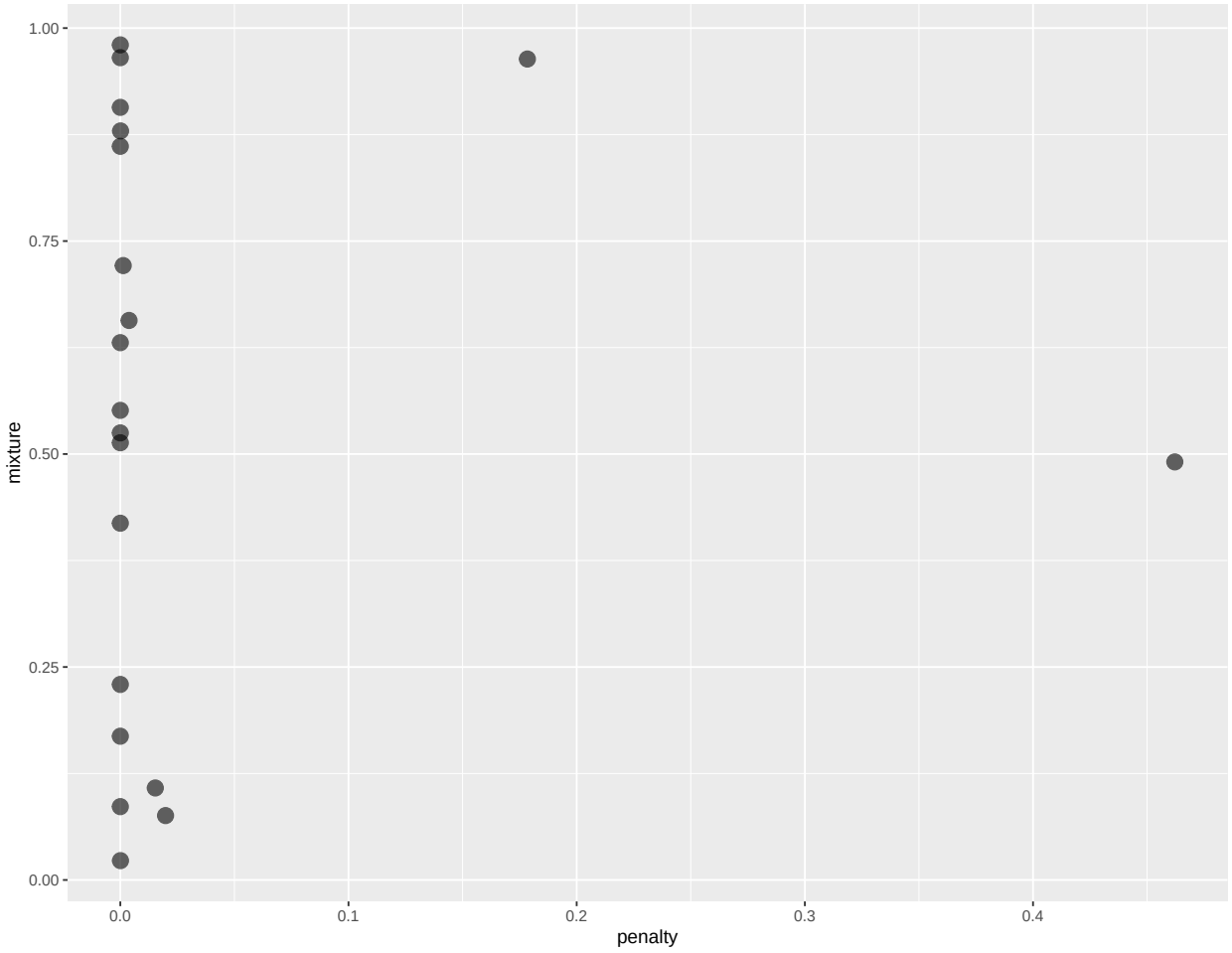
```
ranger_grid %>%
  ggplot(aes(mtry, min_n)) +
  geom_point(size = 4, alpha = 0.6) +
  labs(title = "Ranger: Regular grid for min_n & mtry combinations")
```

```
glm_grid <-
  parameters(glm_model) %>%
  grid_random(size = 20)
glm_grid
```

```
## # A tibble: 20 x 2
##     penalty mixture
##       <dbl>   <dbl>
##  1 1.78e- 1  0.964
##  2 7.95e- 7  0.169
##  3 7.57e- 8  0.0861
##  4 8.08e-10  0.861
##  5 1.66e- 7  0.525
##  6 3.83e- 3  0.657
##  7 1.11e- 9  0.230
##  8 1.27e- 3  0.721
##  9 4.62e- 1  0.491
## 10 1.03e- 8  0.965
## 11 1.22e- 9  0.907
## 12 3.58e-10  0.551
## 13 1.98e- 2  0.0756
## 14 6.49e- 5  0.0227
## 15 5.02e- 6  0.513
## 16 4.47e- 7  0.631
## 17 6.33e- 8  0.419
## 18 9.93e- 5  0.879
## 19 1.53e- 2  0.108
## 20 9.51e-10  0.980
```

```
glm_grid %>% ggplot(aes(penalty, mixture)) +
  geom_point(size = 4, alpha = 0.6) +
  labs(title = "GLM: Random grid for penalty & mixture combinations")
```

GLM: Random grid for penalty & mixture combinations

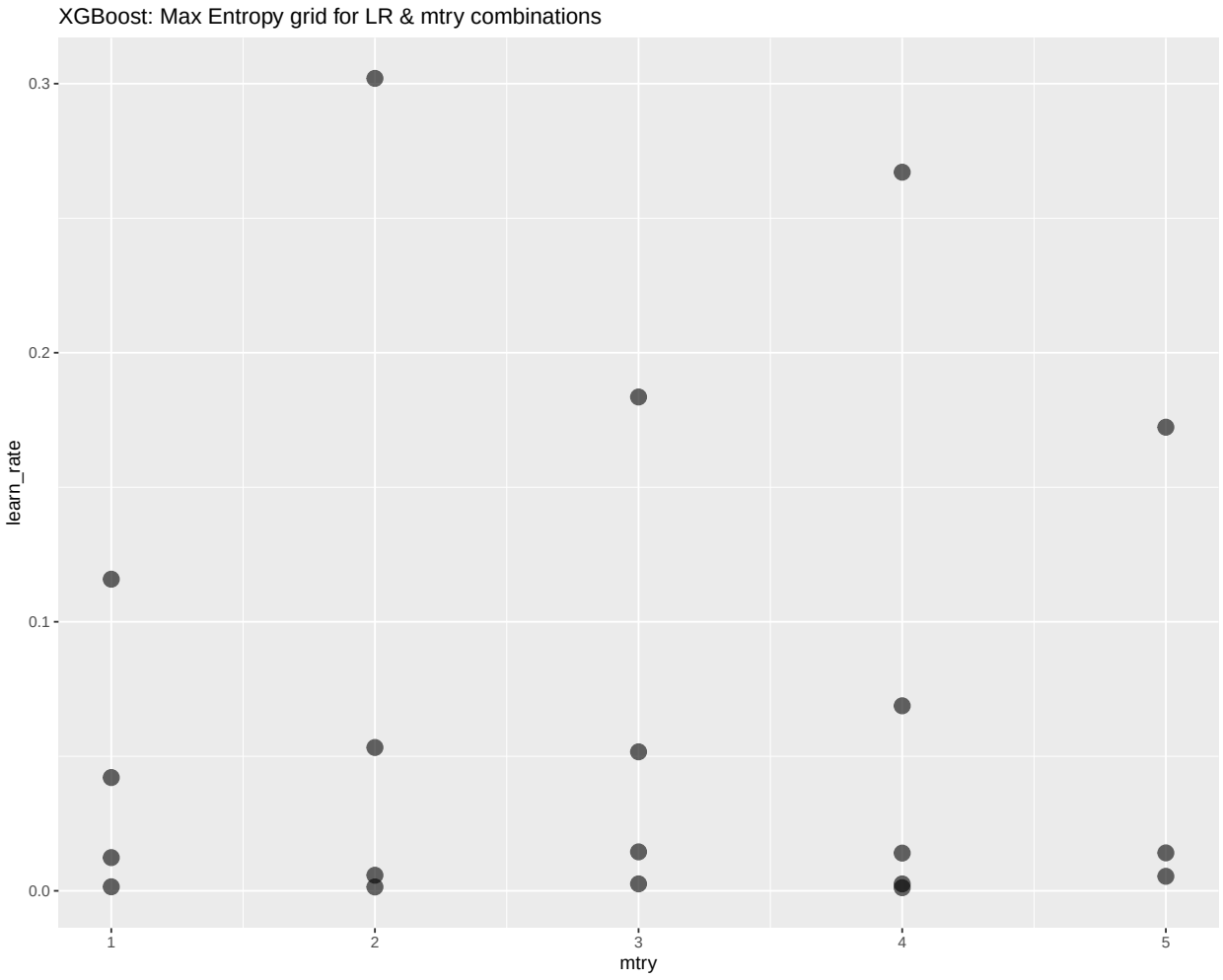



```
xgboost_grid <-
  parameters(xgboost_model) %>%
  finalize(select(training(penguins_split), -species)) %>%
  grid_max_entropy(size = 20)
xgboost_grid
```

```
## # A tibble: 20 x 2
##     mtry learn_rate
##    <int>      <dbl>
##  1     3    0.00259
##  2     4    0.0140
##  3     4    0.00257
##  4     3    0.0145
##  5     5    0.0141
##  6     3    0.184
##  7     2    0.00580
##  8     4    0.267
##  9     1    0.116
## 10     4    0.0688
## 11     4    0.00117
## 12     5    0.00539
## 13     1    0.00150
```

```
## 14     2    0.302
## 15     1    0.0421
## 16     5    0.172
## 17     1    0.0123
## 18     3    0.0517
## 19     2    0.0533
## 20     2    0.00144
```

```
xgboost_grid %>% ggplot(aes(mtry, learn_rate)) +
  geom_point(size = 4, alpha = 0.6) +
  labs(title = "XGBoost: Max Entropy grid for LR & mtry combinations")
```



1.7 데이터 전처리

```
recipe_base <-
  recipe(species ~ ., data = training(penguins_split)) %>%
  step_dummy(all_nominal(), -all_outcomes(), one_hot = TRUE) # Create dummy variables (which glmnet nee

recipe_1 <-
  recipe_base %>%
  step_YeoJohnson(all_numeric())
```

```
recipe_1 %>%
  prep() %>%
  juice() %>%
  summary()
```

```
##  bill_length_mm  bill_depth_mm   flipper_length_mm  body_mass_g
##  Min.   :14.23   Min.   :33.71   Min.   :0.5141    Min.   :1.780
##  1st Qu.:16.34   1st Qu.:43.35   1st Qu.:0.5141    1st Qu.:1.783
##  Median :17.97   Median :49.99   Median :0.5141    Median :1.784
##  Mean   :17.78   Mean   :49.69   Mean   :0.5141    Mean   :1.784
##  3rd Qu.:19.12   3rd Qu.:55.86   3rd Qu.:0.5141    3rd Qu.:1.786
##  Max.   :22.14   Max.   :68.21   Max.   :0.5141    Max.   :1.788
##       species      sex_female        sex_male
##  Adelie   :102   Min.   :0.0000   Min.   :0.0000
##  Chinstrap: 47   1st Qu.:0.0000   1st Qu.:0.0000
##  Gentoo   : 83   Median :0.0000   Median :1.0000
##                  Mean   :0.4957   Mean   :0.5043
##                  3rd Qu.:1.0000   3rd Qu.:1.0000
##                  Max.   :1.0000   Max.   :1.0000
```

```
recipe_2 <-
  recipe_base %>%
  step_normalize(all_numeric())

recipe_2 %>%
  prep() %>%
  juice() %>%
  summary()
```

```
##  bill_length_mm     bill_depth_mm     flipper_length_mm  body_mass_g
##  Min.   :-2.11983   Min.   :-2.0693   Min.   :-2.0300   Min.   :-1.6758
##  1st Qu.:-0.89908   1st Qu.:-0.7748   1st Qu.:-0.7647   1st Qu.:-0.8078
##  Median : 0.09727   Median : 0.0628   Median :-0.2727   Median :-0.1879
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.82433   3rd Qu.: 0.7735   3rd Qu.: 0.9223   3rd Qu.: 0.6879
##  Max.   : 2.81703   Max.   : 2.1949   Max.   : 2.1173   Max.   : 2.6020
##       species      sex_female         sex_male
##  Adelie   :102   Min.   :-0.9893   Min.   :-1.0065
##  Chinstrap: 47   1st Qu.:-0.9893   1st Qu.:-1.0065
##  Gentoo   : 83   Median :-0.9893   Median : 0.9893
##                  Mean   : 0.0000   Mean   : 0.0000
##                  3rd Qu.: 1.0065   3rd Qu.: 0.9893
##                  Max.   : 1.0065   Max.   : 0.9893
```

1.8 Metrics

```
model_metrics <- yardstick::metric_set(f_meas, pr_auc)
```

1.9 K-Fold CV

```
data_penguins_3_cv_folds <-
  rsample::vfold_cv(
    v = 5,
```

```
    data = training(penguins_split),
    strata = species
  )
```

1.10 모델 학습(Model Training)

1.10.1 일괄 작업 작성

```
ranger_r1_workflow <-
  workflows::workflow() %>%
  add_model(ranger_model) %>%
  add_recipe(recipe_1)

glm_r2_workflow <-
  workflows::workflow() %>%
  add_model(glm_model) %>%
  add_recipe(recipe_2)

xgboost_r2_workflow <-
  workflows::workflow() %>%
  add_model(xgboost_model) %>%
  add_recipe(recipe_2)
```

1.10.2 Gridsearch를 활용한 학습

```
tic("2. Ranger tune grid training duration ")
ranger_tuned <-
  tune::tune_grid(
    object = ranger_r1_workflow,
    resamples = data_penguins_3_cv_folds,
    grid = ranger_grid,
    metrics = model_metrics,
    control = tune::control_grid(save_pred = TRUE)
  )
toc(log = TRUE)
```

```
## 2. Ranger tune grid training duration : 4.38 sec elapsed
```

```
tic("3. GLM tune grid training duration ")
glm_tuned <-
  tune::tune_grid(
    object = glm_r2_workflow,
    resamples = data_penguins_3_cv_folds,
    grid = glm_grid,
    metrics = model_metrics,
    control = tune::control_grid(save_pred = TRUE)
  )
toc(log = TRUE)
```

```
## 3. GLM tune grid training duration : 4.77 sec elapsed
```

```
tic("4. XGBoost tune grid training duration ")
xgboost_tuned <-
  tune::tune_grid(
```

```
    object = xgboost_r2_workflow,
    resamples = data_penguins_3_cv_folds,
    grid = xgboost_grid,
    metrics = model_metrics,
    control = tune::control_grid(save_pred = TRUE)
  )
toc(log = TRUE)
```

```
## 4. XGBoost tune grid training duration : 4.09 sec elapsed
```

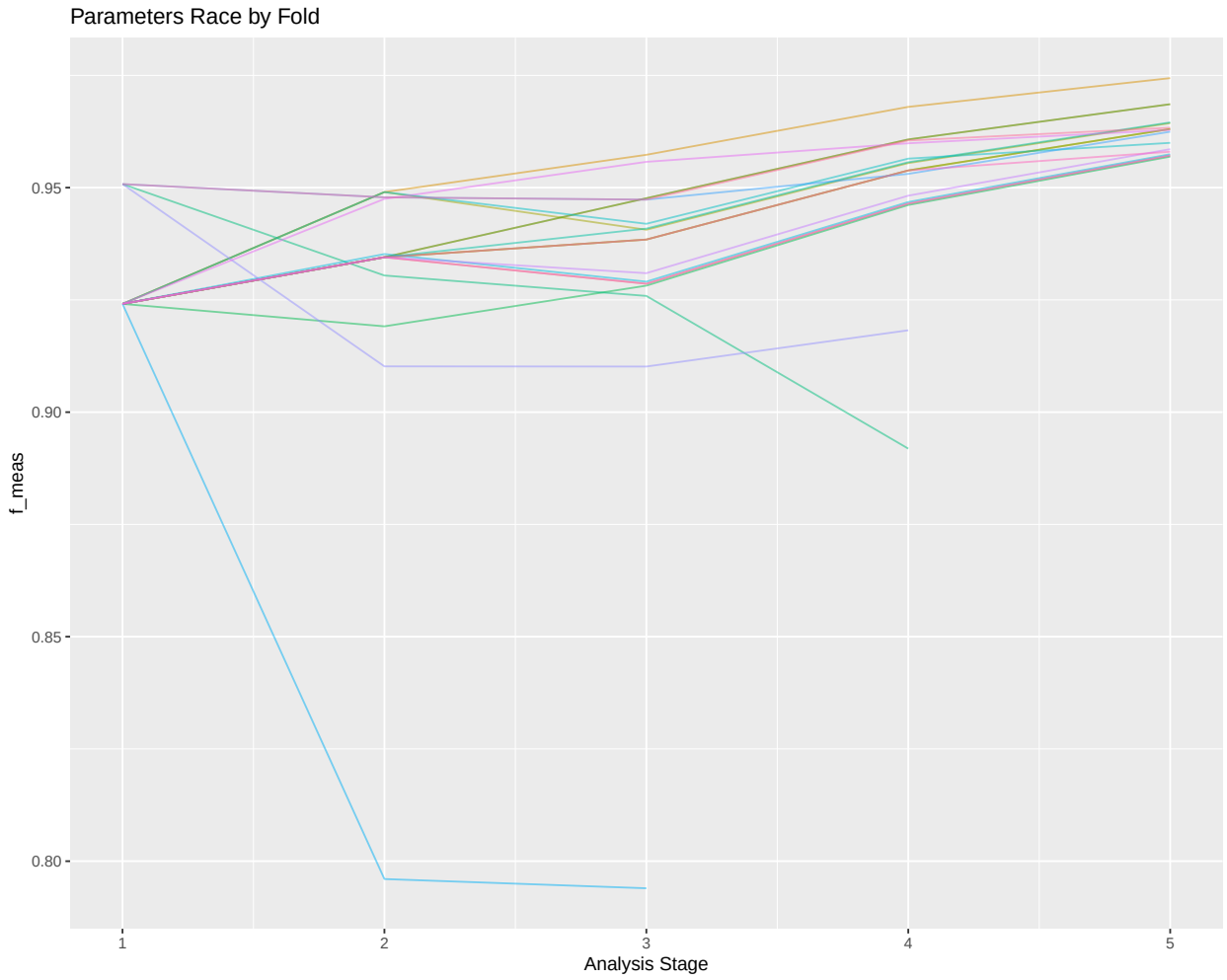
1.10.3 학습 결과 확인

```
tic("5. Tune race training duration ")
ft_xgboost_tuned <-
  finetune::tune_race_anova(
    object = xgboost_r2_workflow,
    resamples = data_penguins_3_cv_folds,
    grid = xgboost_grid,
    metrics = model_metrics,
    control = control_race(verbose_elim = TRUE) # 66
  )
toc(log = TRUE)
```

```
## 5. Tune race training duration : 6.73 sec elapsed
```

1.10.4 시각화를 통한 확인

```
plot_race(ft_xgboost_tuned) + labs(title = "Parameters Race by Fold")
```



1.10.5 결과

```
bind_cols(
  tibble(model = c("Ranger", "GLM", "XGBoost")),
  bind_rows(
    ranger_tuned %>%
      collect_metrics() %>% group_by(.metric) %>% summarise(best_va = max(mean, na.rm = TRUE)) %>% arra
    glm_tuned %>%
      collect_metrics() %>% group_by(.metric) %>% summarise(best_va = max(mean, na.rm = TRUE)) %>% arra
    xgboost_tuned %>%
      collect_metrics() %>% group_by(.metric) %>% summarise(best_va = max(mean, na.rm = TRUE)) %>% arra
  )
)
```

```
## # A tibble: 3 x 3
##   model   f_meas pr_auc
##   <chr>    <dbl>  <dbl>
## 1 Ranger   0.979  0.994
## 2 GLM      0.995  1
## 3 XGBoost  0.974  0.992
```

1.11 전체 모델 확인

```
glm_tuned %>% collect_metrics() # 20 models and 2 metrics
```

```
## # A tibble: 40 x 8
##         penalty mixture .metric .estimator  mean     n  std_err .config
##           <dbl>   <dbl> <chr>   <chr>      <dbl> <int>    <dbl> <chr>
##  1 0.0000649     0.0227 f_meas  macro      0.995     5 0.00506  Preprocessor1_M~
##  2 0.0000649     0.0227 pr_auc  macro      0.999     5 0.000784 Preprocessor1_M~
##  3 0.0198        0.0756 f_meas  macro      0.984     5 0.00641  Preprocessor1_M~
##  4 0.0198        0.0756 pr_auc  macro      0.999     5 0.000784 Preprocessor1_M~
##  5 0.0000000757  0.0861 f_meas  macro      0.995     5 0.00506  Preprocessor1_M~
##  6 0.0000000757  0.0861 pr_auc  macro      1         5 0        Preprocessor1_M~
##  7 0.0153        0.108  f_meas  macro      0.984     5 0.00641  Preprocessor1_M~
##  8 0.0153        0.108  pr_auc  macro      0.999     5 0.000784 Preprocessor1_M~
##  9 0.000000795   0.169  f_meas  macro      0.995     5 0.00506  Preprocessor1_M~
## 10 0.000000795   0.169  pr_auc  macro      1         5 0        Preprocessor1_M~
## # i 30 more rows
```

```
glm_tuned %>%
  collect_metrics() %>%
  group_by(.metric) %>%
  summarise(best_va = max(mean, na.rm = TRUE)) %>%
  arrange(.metric)
```

```
## # A tibble: 2 x 2
##   .metric best_va
##   <chr>     <dbl>
## 1 f_meas    0.995
## 2 pr_auc    1
```

```
glm_tuned %>%
  collect_metrics() %>%
  group_by(.metric) %>%
  summarise(best_va = max(mean, na.rm = TRUE)) %>%
  arrange(.metric)
```

```
## # A tibble: 2 x 2
##   .metric best_va
##   <chr>     <dbl>
## 1 f_meas    0.995
## 2 pr_auc    1
```

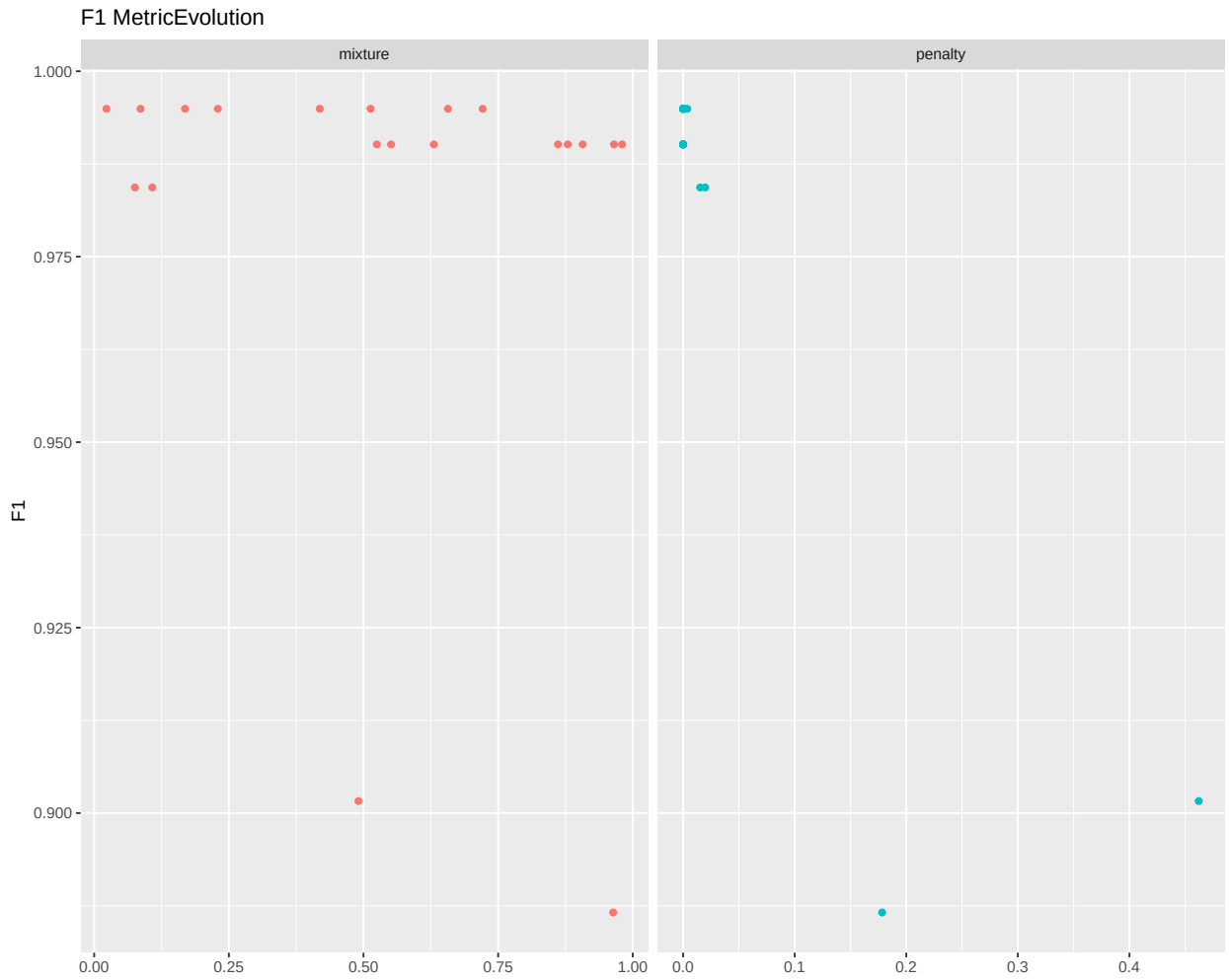
```
glm_tuned %>% select_best(metric = "f_meas")
```

```
## # A tibble: 1 x 3
##     penalty mixture .config
##       <dbl>   <dbl> <chr>
## 1 0.0000649  0.0227 Preprocessor1_Model01
```

1.12 F1값 확인

```
glm_tuned %>%
  collect_metrics() %>%
  filter(.metric == "f_meas") %>%
  select(mean, penalty, mixture) %>%
```

```
  pivot_longer(penalty:mixture,
    values_to = "value",
    names_to = "parameter"
  ) %>%
  ggplot(aes(value, mean, color = parameter)) +
  geom_point(show.legend = FALSE) +
  facet_wrap(~parameter, scales = "free_x") +
  labs(x = NULL, y = "F1", title = "F1 MetricEvolution")
```



1.13 모델 계선

```
best_f1 <-
  select_best(xgboost_tuned, metric = "f_meas")

final_model_op1 <-
  finalize_workflow(
    x = xgboost_r2_workflow,
    parameters = best_f1
  )
```

```
final_model_op1
```

```
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: boost_tree()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_dummy()
## * step_normalize()
##
## -- Model ----------------------------------------------------------------------
## Boosted Tree Model Specification (classification)
##
## Main Arguments:
##   mtry = 3
##   learn_rate = 0.014484928526913
##
## Computational engine: xgboost
```

1.14 Last Fit Tune Model

```
tic("6. Train final model Tune")
penguins_last_fit <-
  last_fit(final_model_op1,
    penguins_split,
    metrics = model_metrics
  )
toc(log = TRUE)
```

```
## 6. Train final model Tune: 0.11 sec elapsed
```

```
collect_metrics(penguins_last_fit) %>%
  arrange(.metric)
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 f_meas  macro              1 Preprocessor1_Model1
## 2 pr_auc  macro              1 Preprocessor1_Model1
```

```
penguins_last_fit %>%
  collect_predictions() %>%
  conf_mat(truth = species, estimate = .pred_class)
```

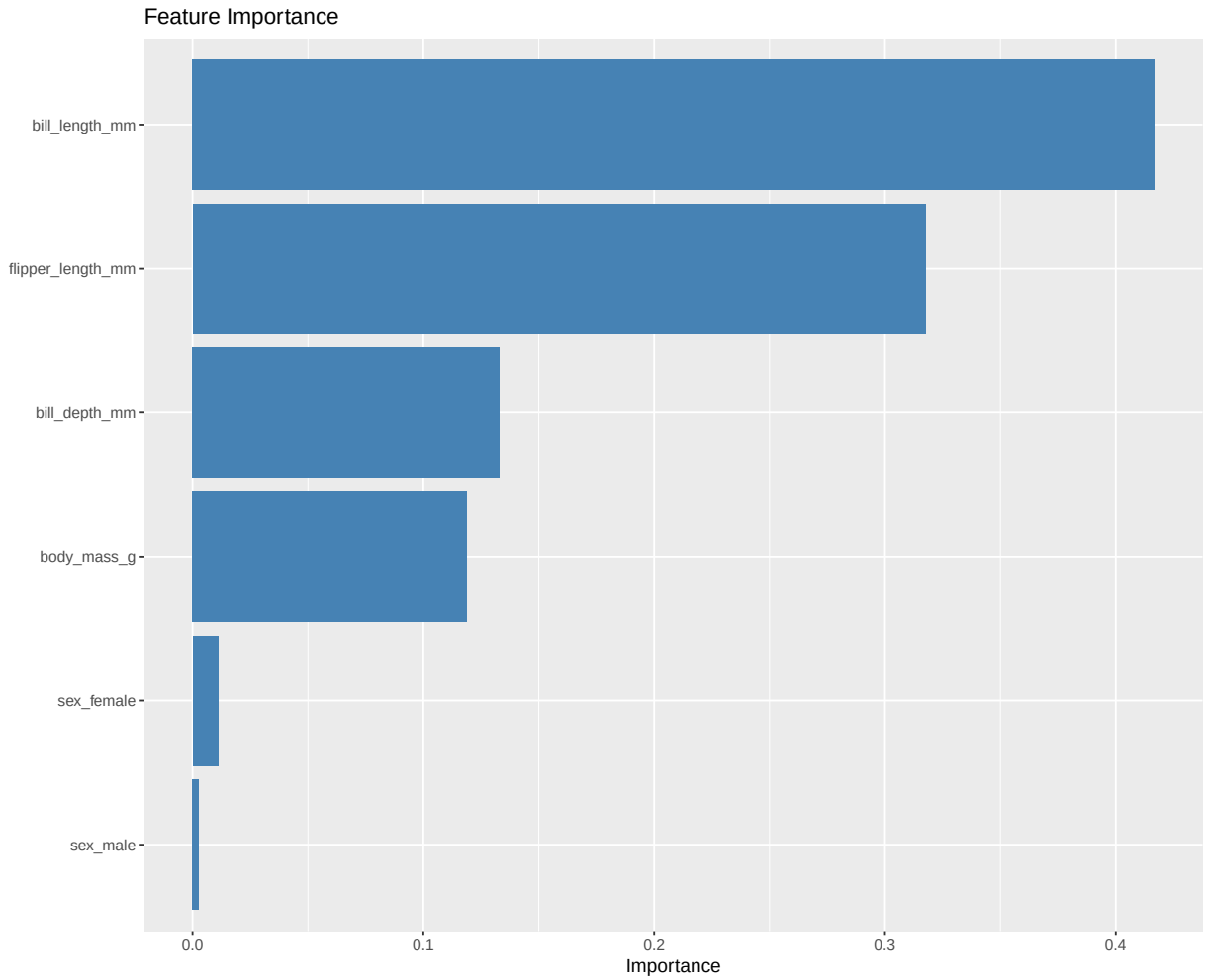
```
##            Truth
## Prediction  Adelie Chinstrap Gentoo
##   Adelie        44         0      0
##   Chinstrap      0        21      0
##   Gentoo         0         0     36
```

```
penguins_last_fit %>%
  pull(.predictions) %>%
  as.data.frame() %>%
  filter(.pred_class != species)
```

```
## [1] .pred_class     .row            .pred_Adelie    .pred_Chinstrap
## [5] .pred_Gentoo    species         .config
## <0 rows> (or 0-length row.names)
```

1.15 주요 특징 분석

```
final_model_op1 %>%
  fit(data = penguins_df) %>%
  pull_workflow_fit() %>%
  vip(
    geom = "col",
    aesthetics = list(fill = "steelblue")
  ) +
  labs(title = "Feature Importance")
```



1.16 모델별 지표 확인

```
tic.log() %>%
  unlist() %>%
```

```
  tibble()
```

```
## # A tibble: 5 x 1
##   .
##   <chr>
## 1 2. Ranger tune grid training duration : 4.38 sec elapsed
## 2 3. GLM tune grid training duration : 4.77 sec elapsed
## 3 4. XGBoost tune grid training duration : 4.09 sec elapsed
## 4 5. Tune race training duration : 6.73 sec elapsed
## 5 6. Train final model Tune: 0.11 sec elapsed
```