XGBOOST FIT

Victor M. Uribe

2023-01-05

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.1     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.1     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 1.0.0 --
## v broom        1.0.4     v rsample      1.1.1
## v dials        1.1.0     v tune         1.0.1
## v infer        1.0.4     v workflows    1.1.3
## v modeldata    1.1.0     v workflowsets 1.0.0
## v parsnip      1.0.4     v yardstick    1.1.0
## v recipes      1.0.5
## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/
```

```
library(finetune)
```

```
members <- data.table::fread('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/

everest <- members %>% # should have an age
  filter(age != 'NA' & peak_name == "Everest" & expedition_role == "Climber") %>%
  select(-c(peak_id,peak_name,expedition_id, member_id, expedition_role, death_height_metres,
            injury_height_metres, highpoint_metres, death_cause, injury_type, hired, solo)) %>%
  mutate(
    #death_height_metres = ifelse(death_height_metres == NA, 0, death_height_metres), # issue with lev
    #injury_height_metres = ifelse(injury_height_metres == NA, 0, injury_height_metres), # gives leveli
    #highpoint_metres = ifelse(highpoint_metres == 'NA', 0, highpoint_metres), # thows off the p values
    season = factor(season),
```

```
    sex = factor(sex),
    citizenship = factor(citizenship), # not significant after testing
    #expedition_role = factor(expedition_role), # not significant after testing
    #hired = factor(hired),
    success = factor(success), # value being predicted
    #solo = factor(solo),
    oxygen_used = factor(oxygen_used),
    died = factor(died),
    #death_cause = factor(death_cause), # issue
    injured = factor(injured)#,
    #injury_type = factor(injury_type) # issue with levels
  )
```

```
everest$success %>% table()
```

```
## .
## FALSE  TRUE
##  6656  3811
```

```
## Original Data Split
set.seed(123)

everest_split <- initial_split(everest, prop = .7)
everest_train <- training(everest_split)
everest_test <- testing(everest_split)

everest_fold <- vfold_cv(everest_train, strata = success)
```

```
xgb_spec <- boost_tree(
  mtry = tune(),
  min_n = tune(),
  trees = tune(),
  #tree_depth = tune(),
  loss_reduction = tune(),
  learn_rate = tune()
) %>%
  set_engine("xgboost", objective = "binary:hinge") %>%
  set_mode("classification")


xgb_rec <- recipe(success ~., everest_train) %>%
  step_other(citizenship, threshold = .05) %>%
  step_dummy(all_nominal_predictors())


xgb_wf <- workflow() %>%
  add_model(xgb_spec) %>%
  add_recipe(xgb_rec)
```

```
doParallel::registerDoParallel()
set.seed(123)

xgb_para <- parameters(
  finalize(mtry(), everest_train),
  trees(),
```

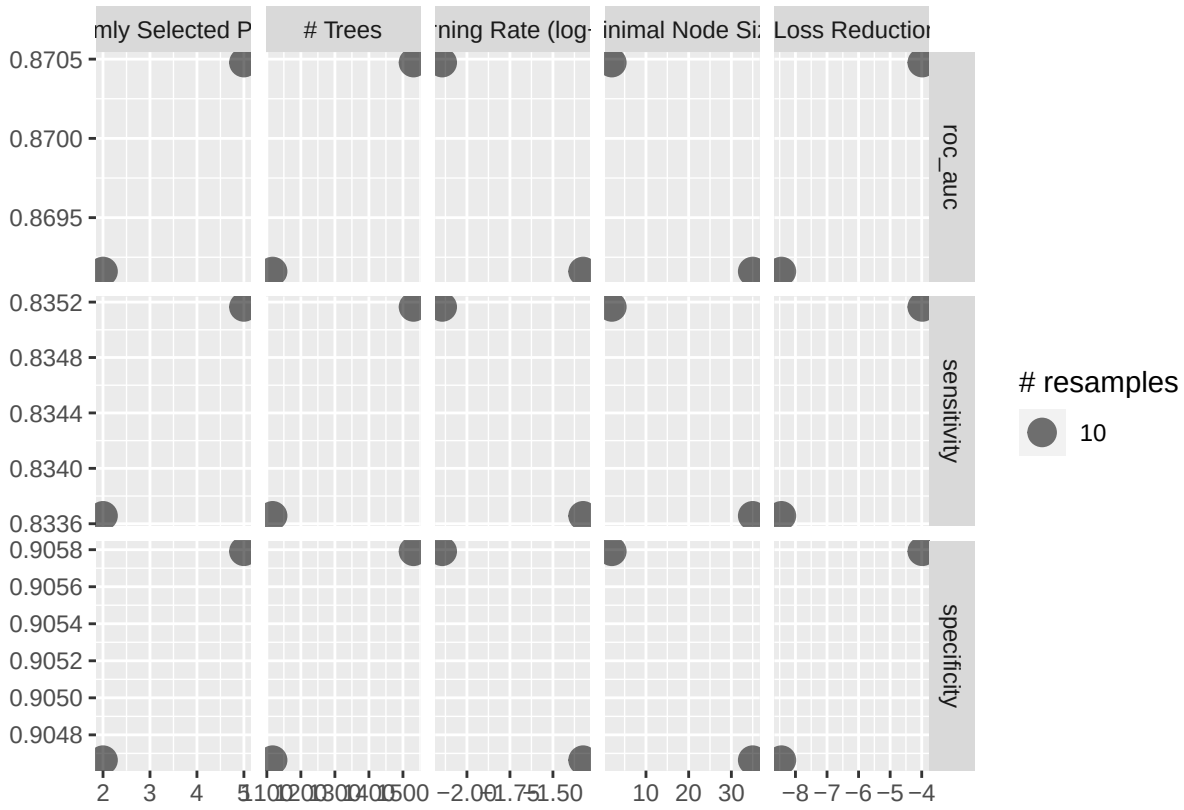
```
  loss_reduction(),
  learn_rate(),
  min_n()
  #tree_depth
  #finalize(sample_size(), everest_train)
)

xgb_grid <- grid_max_entropy(
  xgb_para,
  size = 20
)

xgb_tune <- tune_race_anova(
  xgb_wf,
  resamples = everest_fold,
  grid = xgb_grid,
  metrics = metric_set(roc_auc, sensitivity, specificity),
  control = control_race(verbose_elim = TRUE)
)
```

```
## i Racing will maximize the roc_auc metric.
## i Resamples are analyzed in a random order.
## i Fold10: 16 eliminated;  4 candidates remain.
## i Fold06:  2 eliminated;  2 candidates remain.
## i Fold04:  0 eliminated;  2 candidates remain.
## i Fold05:  0 eliminated;  2 candidates remain.
## i Fold07:  0 eliminated;  2 candidates remain.
## i Fold03:  0 eliminated;  2 candidates remain.
## i Fold02:  0 eliminated;  2 candidates remain.
```

```
xgb_tune %>% autoplot()
```




```
doParallel::registerDoParallel()
best_xgb <- xgb_tune %>%
  select_best("roc_auc")

fin_xgb_wf <- finalize_workflow(
  xgb_wf,
  best_xgb
)

fit_xgb <- fin_xgb_wf %>%
  fit(everest_train)
```

```
doParallel::registerDoParallel()

xgb_pred <- fit_xgb %>%
  predict(new_data = everest_test)

xgb_tmp <- factor(ifelse(xgb_pred == TRUE, TRUE, FALSE))
caret::confusionMatrix(xgb_tmp, everest_test[["success"]])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1659   94
##      TRUE    356 1032
##
##                Accuracy : 0.8567
##                  95% CI : (0.844, 0.8688)
```

```
##     No Information Rate : 0.6415
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7037
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8233
##             Specificity : 0.9165
##          Pos Pred Value : 0.9464
##          Neg Pred Value : 0.7435
##              Prevalence : 0.6415
##          Detection Rate : 0.5282
##    Detection Prevalence : 0.5581
##       Balanced Accuracy : 0.8699
##
##        'Positive' Class : FALSE
##
```

Accuracy : 0.8571 Sensitivity : 0.8303
Specificity : 0.9050

Collapsing Categories to just show as "other" for all

```
#new_recipe <- recipe(success~., everest_train) %>%
#  step_other(citizenship, threshold = .7) %>%
#  step_dummy(all_nominal_predictors())

#new_wf <- fin_xgb_wf %>%
#  remove_recipe()

#new_wf <- new_wf %>%
#  add_recipe(new_recipe)

#new_fit <- new_wf %>%
#  fit(everest_train)
```

```
#v2 <- vip::vip(new_fit %>% pull_workflow_fit(), aesthetic = list(fill = "lightblue")) + theme_minimal(
#v2
```

threshold = .15:

- Accuracy: 0.8545
- Sensitivity: 0.8203
- Specificity: 0.9156

threshold = .10:

- Accuracy: 0.8545
- Sensitivity: 0.8203
- Specificity: 0.9156

threshold = .025:

- Accuracy: 0.8564
- Sensitivity: 0.8243

- Specificity: 0.9139

threshold = .05:

- Accuracy: 0.8567
- Sensitivity: 0.8233
- Specificity: 0.9165

threshold = .01:

- Accuracy: 0.8510
- Sensitivity: 0.8164
- Specificity: 0.9130

threshold = .005:

- Accuracy: 0.8459
- Sensitivity: 0.7970
- Specificity: 0.9334

Filtered

```
set.seed(123)

filtered_split <- initial_split(everest_filtered, prop = .7)
filtered_train <- training(filtered_split)
filtered_test <- testing(filtered_split)

filtered_fold <- vfold_cv(filtered_train, strata = success)
```

```
fil_spec <- logistic_reg() %>%
  set_engine("glm")

fil_rec <- recipe(success ~., filtered_train) %>%
  step_other(citizenship, threshold = .5) #%>%
  #step_impute_knn(all_nominal_predictors()) %>%
  #step_dummy(all_nominal_predictors())

fil_wf <- workflow() %>%
  add_model(fil_spec) %>%
  add_recipe(fil_rec)
```

```
doParallel::registerDoParallel()

fil_fit <- fil_wf %>%
  fit(filtered_train)

fil_fit
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_other()
```

```
##
## -- Model ---------------------------------------------------------------------
##
## Call:  stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)
##
## Coefficients:
##      (Intercept)              year      seasonSpring      seasonSummer
##        -88.91879           0.04253           0.74604          -0.67644
##     seasonWinter              sexM               age  citizenshipother
##         -0.92083           0.01051          -0.02118           0.12656
##  oxygen_usedTRUE          diedTRUE       injuredTRUE
##          4.60360          -0.02636          -1.00684
##
## Degrees of Freedom: 7322 Total (i.e. Null);  7312 Residual
## Null Deviance:       9586
## Residual Deviance: 4764  AIC: 4786
```

```
fil_pred <- fil_fit %>%
  predict(filtered_test)

fil_tmp <- factor(ifelse(fil_pred == TRUE, TRUE, FALSE))
caret::confusionMatrix(fil_tmp, filtered_test[["success"]])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1587   77
##      TRUE    393 1082
##
##                Accuracy : 0.8503
##                  95% CI : (0.8373, 0.8626)
##     No Information Rate : 0.6308
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6958
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8015
##             Specificity : 0.9336
##          Pos Pred Value : 0.9537
##          Neg Pred Value : 0.7336
##              Prevalence : 0.6308
##          Detection Rate : 0.5056
##    Detection Prevalence : 0.5301
##       Balanced Accuracy : 0.8675
##
##        'Positive' Class : FALSE
##
```

```
fil_fit %>% extract_fit_engine() %>% car::vif()
```

```
##                 GVIF Df GVIF^(1/(2*Df))
## year        1.340482  1        1.157792
```

```
## season      1.235788  3        1.035915
## sex         1.031022  1        1.015393
## age         1.141841  1        1.068570
## citizenship 1.023596  1        1.011729
## oxygen_used 1.005758  1        1.002875
## died        1.006721  1        1.003355
## injured     1.004650  1        1.002322
```

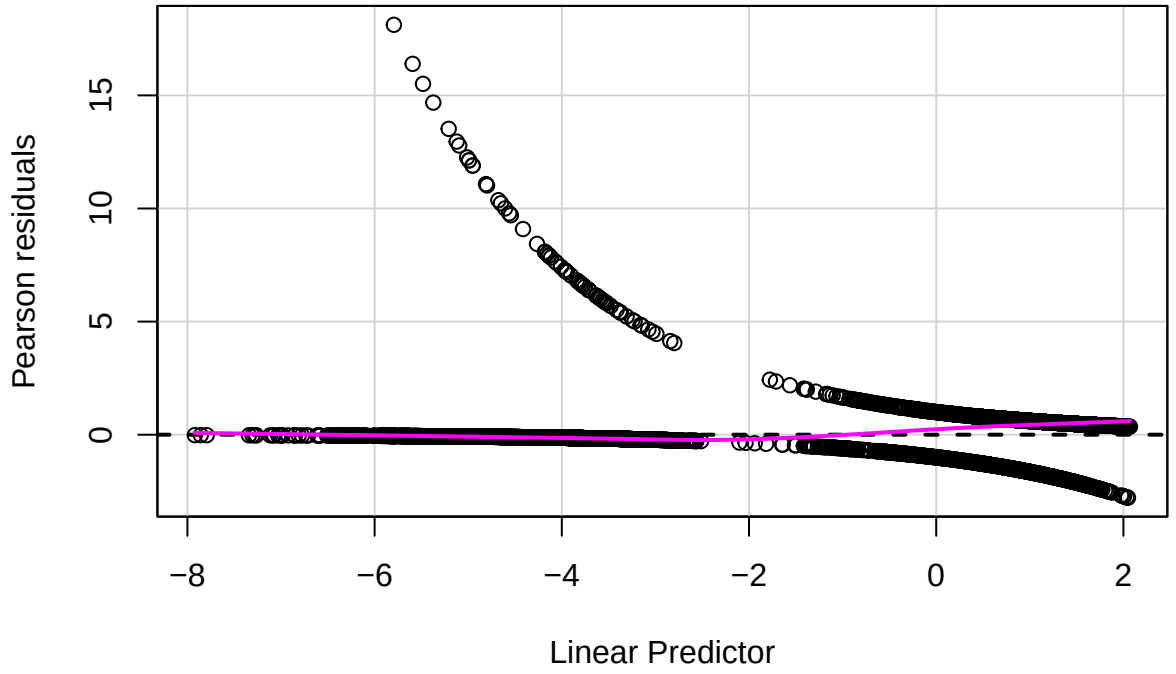
Looking at GVIF we see that there is no multicollinearity present.

```
fil_fit %>% extract_fit_engine() %>% car::durbinWatsonTest()
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1     -0.01474778      2.029025    0.23
##  Alternative hypothesis: rho != 0
```

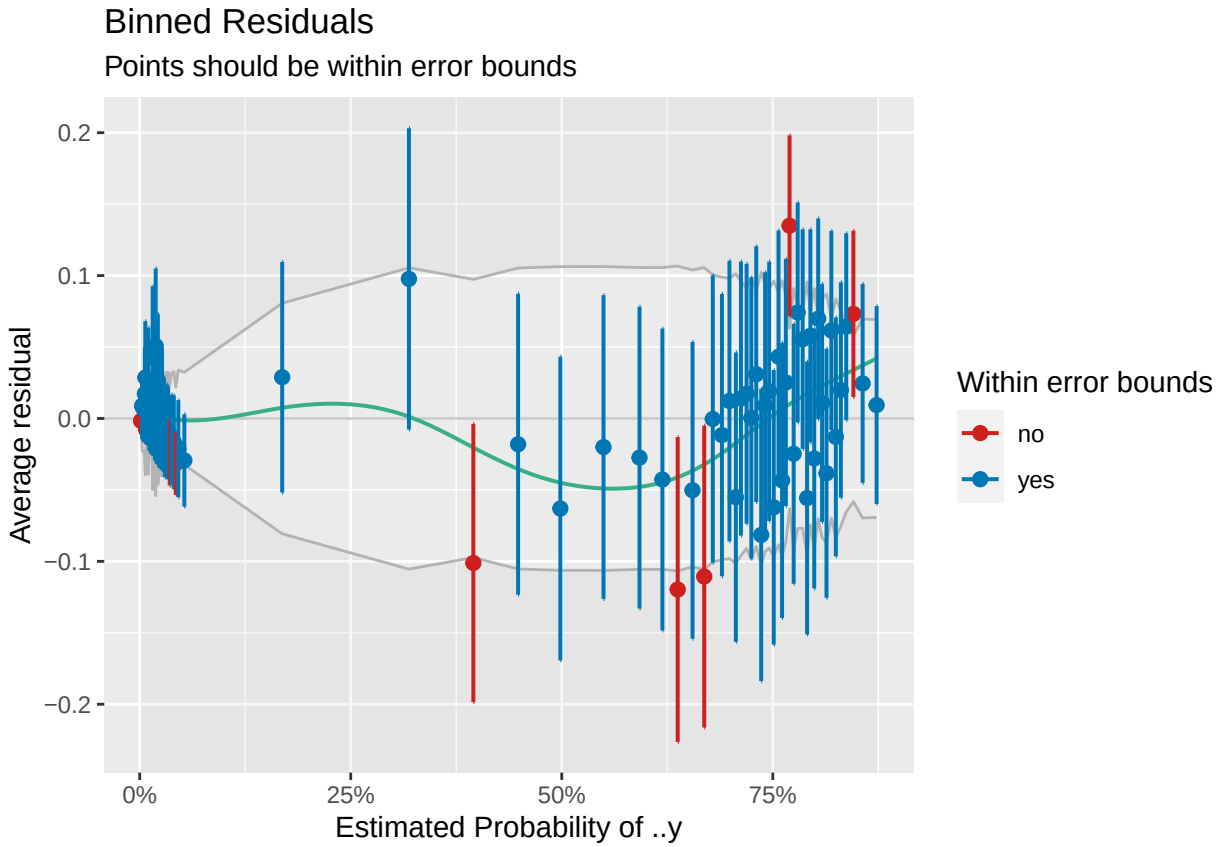
Since the p value is greater than .05 this means that the observations are independent

```
car::residualPlot(fil_fit %>% extract_fit_engine())
```



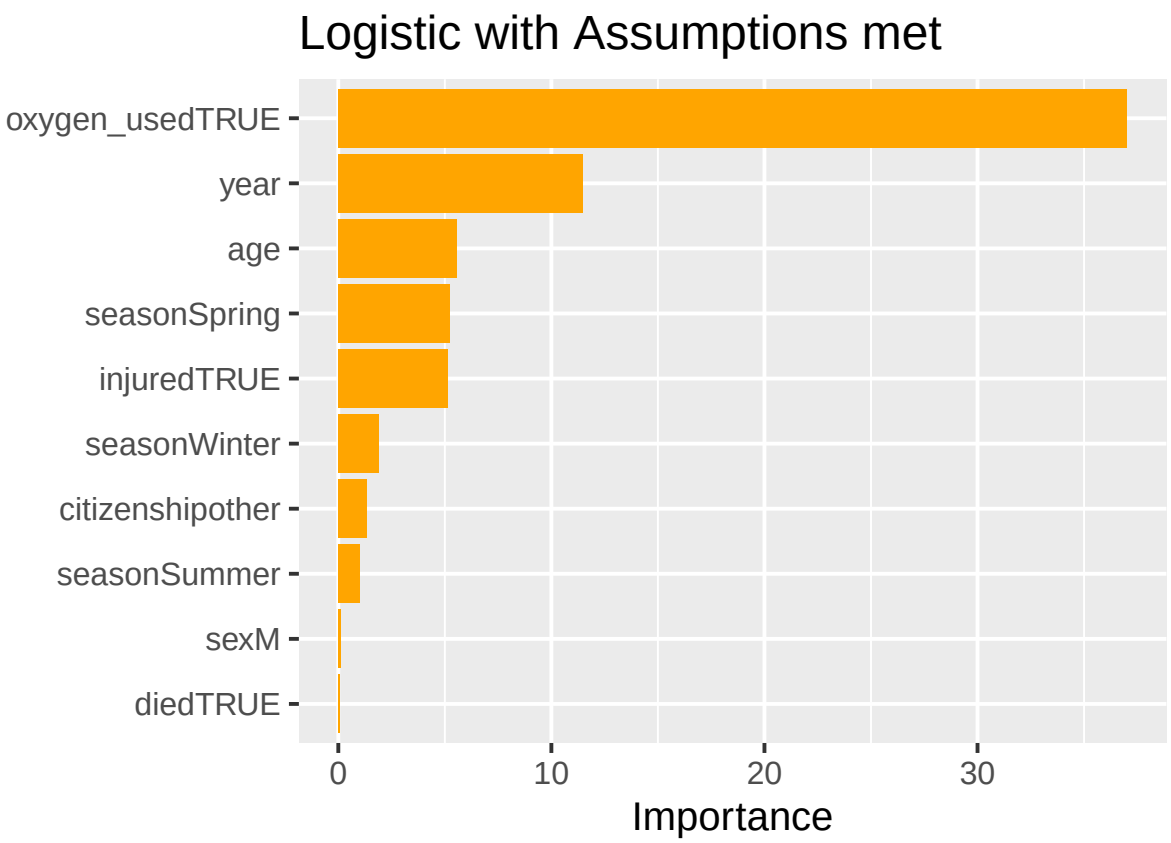
Not much to go off of here

```
plot(performance::binned_residuals(fil_fit %>% extract_fit_engine()))
```



All assumptions are met and we move forward.

```
fil_fit %>% extract_fit_engine() %>% vip::vip(aesthetics = list(fill = "orange")) +
  ggtitle("Logistic with Assumptions met") + theme_gray(base_size = 15)
```



Tuned Model Specs

```
#fin_xgb_wf

#== Workflow ========================================
#Preprocessor: Recipe
#Model: boost_tree()

#== Preprocessor ====================================
#3 Recipe Steps

#• step_rose()
#• step_other()
#• step_dummy()

#== Model ===========================================
#Boosted Tree Model Specification (classification)

#Main Arguments:
#  mtry = 9
#  trees = 1528
#  min_n = 36
#  learn_rate = 0.00501190048405904
#  loss_reduction = 1.03142212215448e-05

#Engine-Specific Arguments:
```

```
#  objective = binary:hinge

#Computational engine: xgboost
```

```
#log_fit

#=== Workflow [trained] ===================================
#Preprocessor: Recipe
#Model: logistic_reg()

#===Preprocessor===========================================
#1 Recipe Step

# step_other()

#== Model ==================================================

#Call:  stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)

#Coefficients:
#     (Intercept)             year      seasonSpring      seasonSummer      #seasonWinter
#       -90.11493          0.04320           0.81122          -0.63855          -0.94062          -0.0
# oxygen_usedTRUE          diedTRUE       injuredTRUE
#         4.59497          0.01795          -1.04636

#Degrees of Freedom: 7325 Total (i.e. Null);  7315 Residual
#Null Deviance:     9627
#Residual Deviance: 4778    AIC: 4800
```

```
#fil_fit

#== Workflow [trained] ====================================
#Preprocessor: Recipe
#Model: logistic_reg()

#== Preprocessor ==========================================
#1 Recipe Step

# step_other()

#== Model ==================================================

#Call:  stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)

#Coefficients:
#     (Intercept)             year      seasonSpring      seasonSummer      seasonWinter
#       -88.91879          0.04253           0.74604          -0.67644          -0.92083           0.0
# oxygen_usedTRUE          diedTRUE       injuredTRUE
#         4.60360          -0.02636          -1.00684

#Degrees of Freedom: 7322 Total (i.e. Null);  7312 Residual
#Null Deviance:     9586
#Residual Deviance: 4764    AIC: 4786
```

```
#fin_ran_wf

#== Workflow ================================================
#Preprocessor: Recipe
#Model: rand_forest()

#== Preprocessor ============================================
#2 Recipe Steps

# step_rose()
# step_other()

#== Model ===================================================
#Random Forest Model Specification (classification)

#Main Arguments:
#  mtry = 3
#  trees = 26
#  min_n = 34

#Engine-Specific Arguments:
#  importance = impurity

#Computational engine: ranger
```