Classifier

```r
set.seed(42)

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- 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
```

```r
library(tidymodels)
```

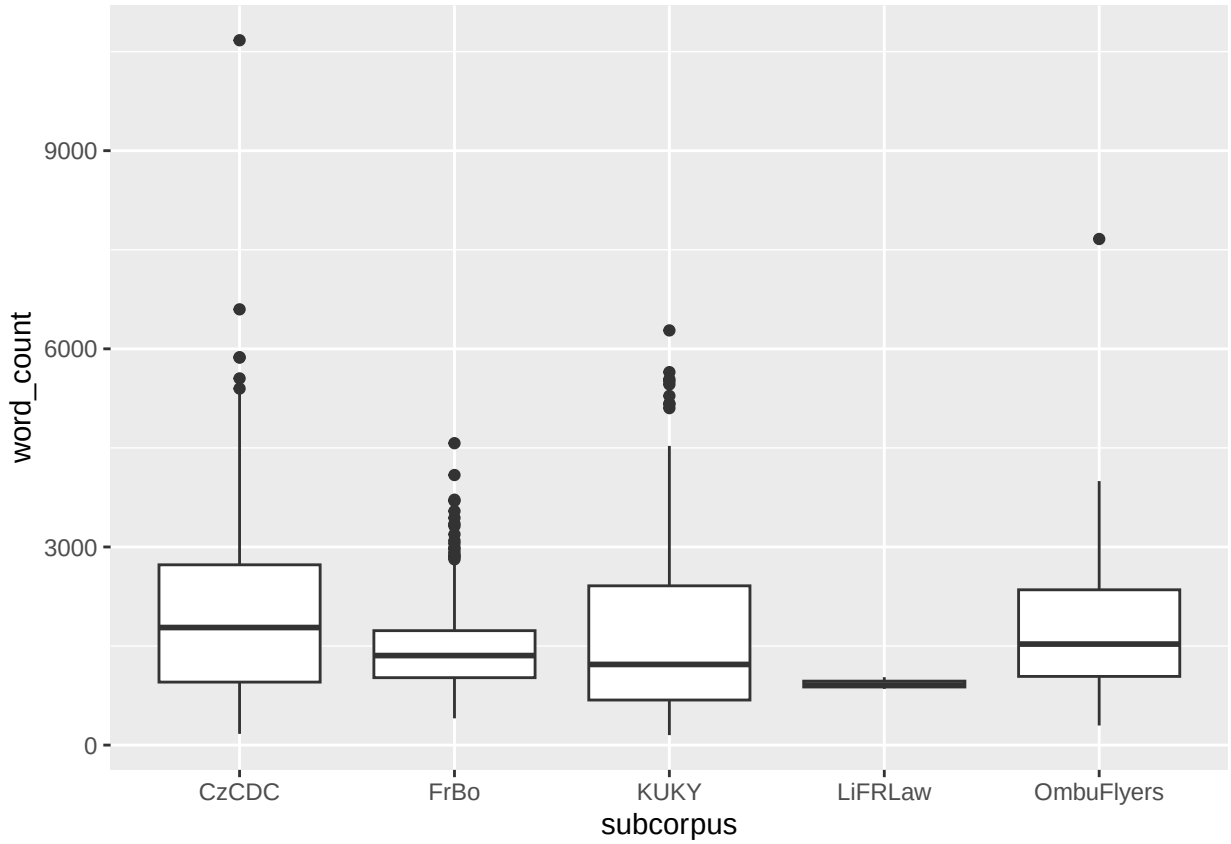
```
## -- Attaching packages --------------------------------------- tidymodels 1.2.0 --
## v broom        1.0.5     v rsample      1.2.1
## v dials        1.3.0     v tune         1.2.1
## v infer        1.0.7     v workflows    1.1.4
## v modeldata    1.4.0     v workflowsets 1.1.0
## v parsnip      1.2.1     v yardstick    1.3.2
## v recipes      1.1.0
## -- 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/
```

Load and tidy data

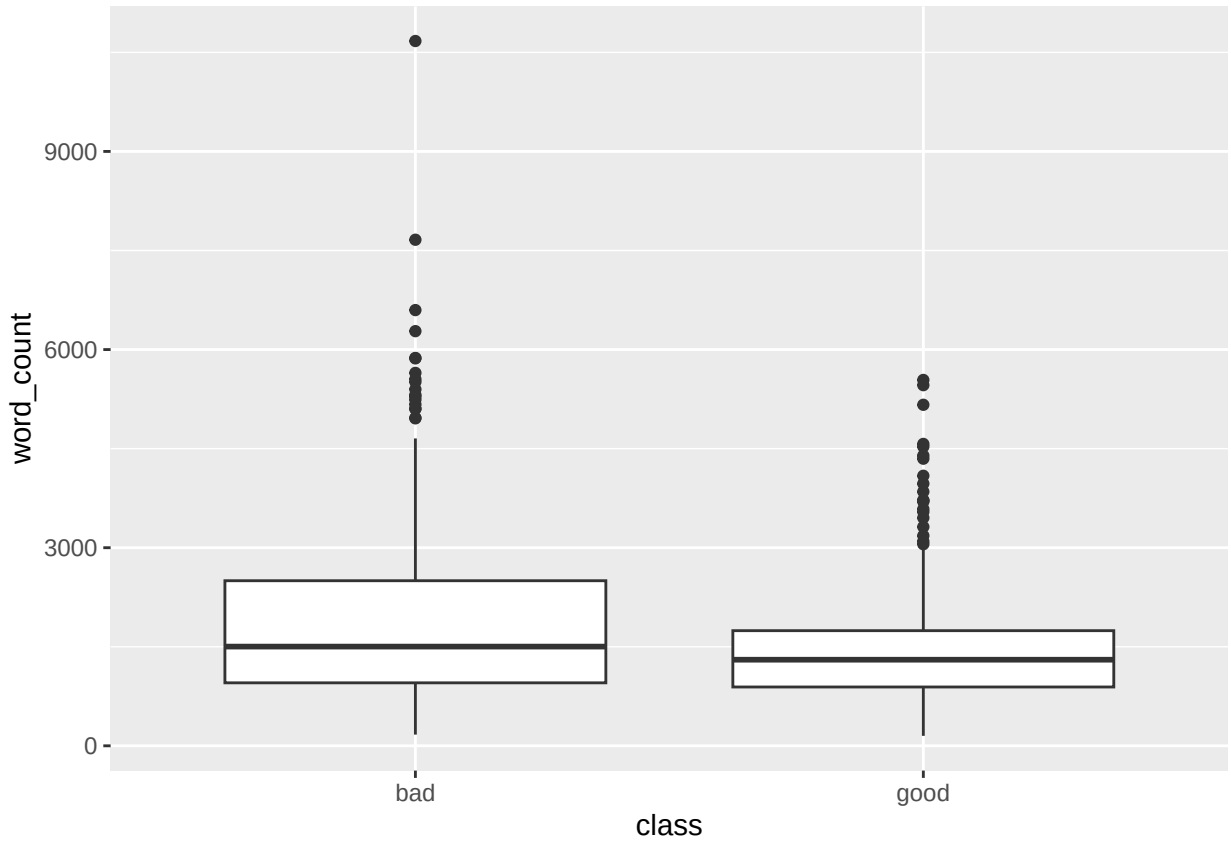
```r
data <- read_csv("../measurements/measurements.csv")
```

```
## Rows: 769 Columns: 96
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (9): fpath, KUK_ID, class, FileName, FolderPath, subcorpus, DocumentTit...
## dbl (85): RuleAbstractNouns, RuleAmbiguousRegards, RuleAnaphoricReferences, ...
## lgl  (2): ClarityPursuit, SyllogismBased
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data %>% ggplot(aes(x = subcorpus, word_count)) +
  geom_boxplot()
```




```
data %>% ggplot(aes(x = class, word_count)) +
  geom_boxplot()
```




```
data_clean <- data %>%
  select(!c(
    fpath,
    KUK_ID,
    FileName,
    FolderPath,
    # subcorpus,
    DocumentTitle,
    ClarityPursuit,
    Readability,
    SyllogismBased,
    SourceDB
  )) %>%
  # replace -1s in variation coefficients with NAs
  mutate(across(c(
    `RuleDoubleAdpos.max_allowable_distance.v`,
    `RuleTooManyNegations.max_negation_frac.v`,
    `RuleTooManyNegations.max_allowable_negations.v`,
    `RuleTooManyNominalConstructions.max_noun_frac.v`,
    `RuleTooManyNominalConstructions.max_allowable_nouns.v`,
    `RuleCaseRepetition.max_repetition_count.v`,
    `RuleCaseRepetition.max_repetition_frac.v`,
    `RulePredSubjDistance.max_distance.v`,
    `RulePredObjDistance.max_distance.v`,
    `RuleInfVerbDistance.max_distance.v`,
    `RuleMultiPartVerbs.max_distance.v`,
    `RuleLongSentences.max_length.v`,
```

```
  `RulePredAtClauseBeginning.max_order.v`,
  `mattr.v`,
  `maentropy.v`
), ~ na_if(.x, -1))) %>%
# replace NAs with 0s
replace_na(list(
  RuleGPcoordovs = 0,
  RuleGPdeverbaddr = 0,
  RuleGPpatinstr = 0,
  RuleGPdeverbsubj = 0,
  RuleGPadjective = 0,
  RuleGPpatbenperson = 0,
  RuleGPwordorder = 0,
  RuleDoubleAdpos = 0,
  RuleDoubleAdpos.max_allowable_distance = 0,
  RuleDoubleAdpos.max_allowable_distance.v = 0,
  RuleAmbiguousRegards = 0,
  RuleReflexivePassWithAnimSubj = 0,
  RuleTooManyNegations = 0,
  RuleTooManyNegations.max_negation_frac = 0,
  RuleTooManyNegations.max_negation_frac.v = 0,
  RuleTooManyNegations.max_allowable_negations = 0,
  RuleTooManyNegations.max_allowable_negations.v = 0,
  RuleTooManyNominalConstructions.max_noun_frac.v = 0,
  RuleTooManyNominalConstructions.max_allowable_nouns.v = 0,
  RuleFunctionWordRepetition = 0,
  RuleCaseRepetition.max_repetition_count.v = 0,
  RuleCaseRepetition.max_repetition_frac.v = 0,
  RuleWeakMeaningWords = 0,
  RuleAbstractNouns = 0,
  RuleRelativisticExpressions = 0,
  RuleConfirmationExpressions = 0,
  RuleRedundantExpressions = 0,
  RuleTooLongExpressions = 0,
  RuleAnaphoricReferences = 0,
  RuleLiteraryStyle = 0,
  RulePassive = 0,
  RulePredSubjDistance = 0,
  RulePredSubjDistance.max_distance = 0,
  RulePredSubjDistance.max_distance.v = 0,
  RulePredObjDistance = 0,
  RulePredObjDistance.max_distance = 0,
  RulePredObjDistance.max_distance.v = 0,
  RuleInfVerbDistance = 0,
  RuleInfVerbDistance.max_distance = 0,
  RuleInfVerbDistance.max_distance.v = 0,
  RuleMultiPartVerbs = 0,
  RuleMultiPartVerbs.max_distance = 0,
  RuleMultiPartVerbs.max_distance.v = 0,
  RuleLongSentences.max_length.v = 0,
  RulePredAtClauseBeginning.max_order.v = 0,
  RuleVerbalNouns = 0,
  RuleDoubleComparison = 0,
```

```
  RuleWrongValencyCase = 0,
  RuleWrongVerbonominalCase = 0,
  RuleIncompleteConjunction = 0
)) %>%
# norm data expected to correlate with text length
mutate(across(c(
  RuleGPcoordovs,
  RuleGPdeverbaddr,
  RuleGPpatinstr,
  RuleGPdeverbsubj,
  RuleGPadjective,
  RuleGPpatbenperson,
  RuleGPwordorder,
  RuleDoubleAdpos,
  RuleAmbiguousRegards,
  RuleFunctionWordRepetition,
  RuleWeakMeaningWords,
  RuleAbstractNouns,
  RuleRelativisticExpressions,
  RuleConfirmationExpressions,
  RuleRedundantExpressions,
  RuleTooLongExpressions,
  RuleAnaphoricReferences,
  RuleLiteraryStyle,
  RulePassive,
  RuleVerbalNouns,
  RuleDoubleComparison,
  RuleWrongValencyCase,
  RuleWrongVerbonominalCase,
  RuleIncompleteConjunction,
  num_hapax,
  RuleReflexivePassWithAnimSubj,
  RuleTooManyNominalConstructions,
  RulePredSubjDistance,
  RuleMultiPartVerbs,
  RulePredAtClauseBeginning
), ~ .x / word_count)) %>%
mutate(across(c(
  RuleTooFewVerbs,
  RuleTooManyNegations,
  RuleCaseRepetition,
  RuleLongSentences,
  RulePredObjDistance,
  RuleInfVerbDistance
), ~ .x / sent_count)) %>%
# remove variables identified as "u counts"
select(!c(
  RuleTooFewVerbs,
  RuleTooManyNegations,
  RuleTooManyNominalConstructions,
  RuleCaseRepetition,
  RuleLongSentences,
  RulePredAtClauseBeginning
```

```
  )) %>%
  unite("strata", c(subcorpus, class), sep = "_", remove = FALSE)

# no NAs should be present now
data_clean[!complete.cases(data_clean), ]
```

```
## # A tibble: 0 x 82
## # i 82 variables: strata <chr>, class <chr>, subcorpus <chr>,
## #   RuleAbstractNouns <dbl>, RuleAmbiguousRegards <dbl>,
## #   RuleAnaphoricReferences <dbl>,
## #   RuleCaseRepetition.max_repetition_count <dbl>,
## #   RuleCaseRepetition.max_repetition_count.v <dbl>,
## #   RuleCaseRepetition.max_repetition_frac <dbl>,
## #   RuleCaseRepetition.max_repetition_frac.v <dbl>, ...
```

```
# use tidymodels::step_corr to remove high-correlating variables
```

Prepare splits and folds

```
# CHECK CONSISTENCY WITH analysis.Rmd

.split_prop <- 4 / 5 # proportion of testing data in the dataset
.no_folds <- 10 # no. of folds in v-fold cross-validation

split <- data_clean %>% initial_split(prop = .split_prop)
training_set <- training(split)
evaluation_set <- testing(split)

folds <- vfold_cv(training_set, v = .no_folds, strata = strata)

print(split)
```

```
## <Training/Testing/Total>
## <615/154/769>
```

```
print(folds)
```

```
## #  10-fold cross-validation using stratification
## # A tibble: 10 x 2
##    splits           id
##    <list>           <chr>
##  1 <split [551/64]> Fold01
##  2 <split [551/64]> Fold02
##  3 <split [553/62]> Fold03
##  4 <split [553/62]> Fold04
##  5 <split [553/62]> Fold05
##  6 <split [554/61]> Fold06
##  7 <split [554/61]> Fold07
##  8 <split [554/61]> Fold08
##  9 <split [556/59]> Fold09
## 10 <split [556/59]> Fold10
```

```
# structure of the training set
table(training_set$subcorpus, training_set$class)
```

```
##
##              bad good
##   CzCDC      172    0
##   FrBo        62  179
##   KUKY        70   89
##   LiFRLaw      1    0
##   OmbuFlyers  42    0
```

```
# structure of the evaluation set
table(evaluation_set$subcorpus, evaluation_set$class)
```

```
##
##              bad good
##   CzCDC       38    0
##   FrBo        17   54
##   KUKY        14   21
##   LiFRLaw      2    0
##   OmbuFlyers   8    0
```

Classifier helpers

Models

```
library(vip)
```

```
##
## Attaching package: 'vip'
```

```
## The following object is masked from 'package:utils':
##
##     vi
```

Null model

```
train_null <- function(recipe, folds) {
  null_workflow <- workflow() %>% add_recipe(recipe)

  null_classification <- null_model() %>%
    set_engine("parsnip") %>%
    set_mode("classification")

  null_rs <- fit_resamples(null_workflow %>% add_model(null_classification), folds)

  cat("Null resamples:\n")
  print(null_rs)

  cat("Null metrics:\n")
  collect_metrics(null_rs) %>% print()

  return(null_rs)
}
```

Lasso

```
train_lasso <- function(recipe, training_set, folds) {
  lasso_tune_spec <- logistic_reg(penalty = tune(), mixture = 1) %>%
    set_mode("classification") %>%
    set_engine("glmnet")

  # cat("Lasso specification for tuning:\n")
  # print(lasso_tune_spec)

  lambda_grid <- grid_regular(penalty(), levels = 30)

  lasso_tune_wf <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(lasso_tune_spec)

  cat("Lasso tune workflow:\n")
  print(lasso_tune_wf)

  lasso_tune_rs <- tune_grid(
    lasso_tune_wf,
    folds,
    grid = lambda_grid,
    control = control_resamples(save_pred = TRUE)
  )

  # cat("Lasso tune resamples:\n")
  # print(lasso_tune_rs)

  cat("Lasso tuning metrics:\n")
  # collect_metrics(lasso_tune_rs) %>% print()
  autoplot(lasso_tune_rs) %>% print()

  lasso_tune_rs %>%
    show_best(metric = "roc_auc") %>%
    print()
  lasso_tune_rs %>%
    show_best(metric = "accuracy") %>%
    print()

  best_accuracy <- lasso_tune_rs %>%
    select_by_one_std_err(metric = "accuracy", -penalty)

  cat("Best accuracy:\n")
  print(best_accuracy)

  final_lasso <- lasso_tune_wf %>% finalize_workflow(best_accuracy)
  cat("Final workflow:\n")
  print(final_lasso)

  fitted_lasso <- fit(final_lasso, training_set)

  cat("Final coefficients:\n")
  fitted_lasso %>%
```

```
    extract_fit_parsnip() %>%
    tidy() %>%
    arrange(estimate) %>%
    print(n = 100)

  return(fitted_lasso)
}
```

SVM

```
train_svm <- function(recipe, training_set, folds) {
  svm_spec <- svm_linear() %>%
    set_mode("classification") %>%
    set_engine("kernlab")
  # cat("SVM specification:\n")
  # print(svm_spec)

  svm_wf <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(svm_spec)
  cat("SVM workflow:\n")
  print(svm_wf)

  svm_rs <- fit_resamples(
    svm_wf,
    folds,
    control = control_resamples(save_pred = TRUE)
  )
  # cat("SVM resamples:\n")
  # print(svm_rs)

  cat("SVM metrics:\n")
  collect_metrics(svm_rs) %>% print()

  svm_rs %>%
    collect_predictions() %>%
    roc_curve(truth = class, .pred_bad) %>%
    autoplot() %>%
    print()

  print("\n")

  svm_rs %>%
    collect_predictions() %>%
    group_by(id) %>%
    roc_curve(truth = class, .pred_bad) %>%
    autoplot() %>%
    print()

  print("\n")

  svm_rs %>%
    conf_mat_resampled(tidy = FALSE) %>%
```

```
    autoplot(type = "heatmap") %>%
    print()

  print("\n")

  final_svm <- fit(svm_wf, training_set)

  return(final_svm)
}

# not sure this works
train_svm_tune <- function(recipe, training_set, folds) {
  svm_tune_spec <- svm_linear(cost = tune()) %>%
    set_mode("classification") %>%
    set_engine("kernlab")

  cat("SVM specification for tuning:\n")
  print(svm_tune_spec)

  lambda_grid <- grid_regular(cost(), levels = 10)
  cat("SVM tuning grid:\n")
  print(lambda_grid)

  svm_tune_wf <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(svm_tune_spec)

  cat("SVM tune workflow:\n")
  print(svm_tune_wf)

  svm_tune_rs <- tune_grid(
    svm_tune_wf,
    folds,
    grid = lambda_grid,
    control = control_resamples(save_pred = TRUE)
  )

  cat("SVM tune resamples:\n")
  print(svm_tune_rs)

  cat("SVM tuning metrics:\n")
  collect_metrics(svm_tune_rs) %>% print()
  autoplot(svm_tune_rs) %>% print()

  svm_tune_rs %>%
    show_best(metric = "roc_auc") %>%
    print()
  svm_tune_rs %>%
    show_best(metric = "accuracy") %>%
    print()

  best_accuracy <- svm_tune_rs %>%
    select_by_one_std_err(metric = "accuracy", -cost)
```

```
  cat("Best ROC AUC:\n")
  print(best_accuracy)

  final_svm <- svm_tune_wf %>% finalize_workflow(best_accuracy)

  cat("Final workflow:\n")
  print(final_svm)

  fitted_svm <- fit(final_svm, training_set)

  return(fitted_svm)
}
```

Random forest

```
train_random_forest <- function(recipe, training_set, folds) {
  rf_spec <- rand_forest(trees = 1000) %>%
    set_mode("classification") %>%
    set_engine("ranger", importance = "impurity")

  # cat("RF specification:\n")
  # print(rf_spec)

  rf_wf <- workflow() %>%
    add_recipe(recipe) %>%
    add_model(rf_spec)

  cat("RF workflow:\n")
  print(rf_wf)

  rf_rs <- fit_resamples(
    rf_wf,
    folds,
    control = control_resamples(save_pred = TRUE)
  )
  # cat("RF resamples:\n")
  # print(rf_rs)

  cat("RF metrics:\n")
  collect_metrics(rf_rs) %>% print()


  rf_rs %>%
    collect_predictions() %>%
    roc_curve(truth = class, .pred_bad) %>%
    autoplot() %>%
    print()

  print("\n")

  rf_rs %>%
    collect_predictions() %>%
    group_by(id) %>%
```

```
    roc_curve(truth = class, .pred_bad) %>%
    autoplot() %>%
    print()

  print("\n")

  rf_rs %>%
    conf_mat_resampled(tidy = FALSE) %>%
    autoplot(type = "heatmap") %>%
    print()

  print("\n")

  return(rf_rs)
}
```

Recipes

```
add_corr_remove_step <- function(recipe, training_set) {
  recipe <- recipe %>% step_corr(all_numeric_predictors(), threshold = .9)

  prep <- recipe %>% prep(training = training_set)
  no <- prep %>%
    tidy() %>%
    filter(type == "corr") %>%
    pull(number)
  prep %>%
    tidy(number = no[[1]]) %>%
    print(n = 200)

  return(recipe)
}
```

All variables

```
formula_all <- class ~
  RuleGPcoordovs +
  RuleGPdeverbaddr +
  RuleGPpatinstr +
  RuleGPdeverbsubj +
  RuleGPadjective +
  RuleGPpatbenperson +
  RuleGPwordorder +
  RuleDoubleAdpos +
  RuleDoubleAdpos.max_allowable_distance +
  RuleDoubleAdpos.max_allowable_distance.v +
  RuleAmbiguousRegards +
  RuleReflexivePassWithAnimSubj +
  # RuleTooFewVerbs +
  RuleTooFewVerbs.min_verb_frac +
  # RuleTooManyNegations +
  RuleTooManyNegations.max_negation_frac +
  RuleTooManyNegations.max_negation_frac.v +
```

```
RuleTooManyNegations.max_allowable_negations +
RuleTooManyNegations.max_allowable_negations.v +
# RuleTooManyNominalConstructions +
RuleTooManyNominalConstructions.max_noun_frac +
RuleTooManyNominalConstructions.max_noun_frac.v +
RuleTooManyNominalConstructions.max_allowable_nouns +
RuleTooManyNominalConstructions.max_allowable_nouns +
RuleFunctionWordRepetition +
# RuleCaseRepetition +
RuleCaseRepetition.max_repetition_count +
RuleCaseRepetition.max_repetition_count.v +
RuleCaseRepetition.max_repetition_frac +
RuleCaseRepetition.max_repetition_frac.v +
RuleWeakMeaningWords +
RuleAbstractNouns +
RuleRelativisticExpressions +
RuleConfirmationExpressions +
RuleRedundantExpressions +
RuleTooLongExpressions +
RuleAnaphoricReferences +
RuleLiteraryStyle +
RulePassive +
RulePredSubjDistance +
RulePredSubjDistance.max_distance +
RulePredSubjDistance.max_distance.v +
RulePredObjDistance +
RulePredObjDistance.max_distance +
RulePredObjDistance.max_distance.v +
RuleInfVerbDistance +
RuleInfVerbDistance.max_distance +
RuleInfVerbDistance.max_distance.v +
RuleMultiPartVerbs +
RuleMultiPartVerbs.max_distance +
RuleMultiPartVerbs.max_distance.v +
# RuleLongSentences +
RuleLongSentences.max_length +
RuleLongSentences.max_length.v +
# RulePredAtClauseBeginning +
RulePredAtClauseBeginning.max_order +
RulePredAtClauseBeginning.max_order.v +
RuleVerbalNouns +
RuleDoubleComparison +
RuleWrongValencyCase +
RuleWrongVerbonominalCase +
RuleIncompleteConjunction +
sent_count +
word_count +
syllab_count +
char_count +
cli +
ari +
num_hapax +
entropy +
```

```
  ttr +
  mattr +
  mattr.v +
  maentropy +
  maentropy.v +
  mamr +
  verb_dist +
  activity +
  hpoint +
  atl +
  fre +
  fkgl +
  gf +
  smog

recipe_all_base <- recipe(
  formula_all,
  data = training_set
)

# without the removal of correlating variables
recipe_all_nocorr <- recipe_all_base %>%
  step_normalize(all_numeric_predictors())
recipe_all_nocorr
```

```
##
## -- Recipe ----------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 77
##
## -- Operations
## * Centering and scaling for: all_numeric_predictors()
```

```
# with the removal of correlating variables
recipe_all <- recipe_all_nocorr %>%
  add_corr_remove_step(training_set = training_set)
```

```
## # A tibble: 10 x 2
##    terms                                  id
##    <chr>                                  <chr>
##  1 RuleCaseRepetition.max_repetition_frac.v corr_2shVT
##  2 char_count                             corr_2shVT
##  3 ari                                    corr_2shVT
##  4 ttr                                    corr_2shVT
##  5 maentropy                              corr_2shVT
##  6 hpoint                                 corr_2shVT
##  7 atl                                    corr_2shVT
##  8 gf                                     corr_2shVT
```

```
##  9 smog                                      corr_2shVT
## 10 word_count                                corr_2shVT
```

```
recipe_all
```

```
##
```

```
## -- Recipe ----------------------------------------------------------------
```

```
##
```

```
## -- Inputs
```

```
## Number of variables by role
```

```
## outcome:    1
## predictor: 77
```

```
##
```

```
## -- Operations
```

```
## * Centering and scaling for: all_numeric_predictors()
```

```
## * Correlation filter on: all_numeric_predictors()
```

Counts

```
formula_counts <- class ~
  RuleGPcoordovs +
  RuleGPdeverbaddr +
  RuleGPpatinstr +
  RuleGPdeverbsubj +
  RuleGPadjective +
  RuleGPpatbenperson +
  RuleGPwordorder +
  RuleDoubleAdpos +
  RuleAmbiguousRegards +
  RuleReflexivePassWithAnimSubj +
  RuleFunctionWordRepetition +
  RuleWeakMeaningWords +
  RuleAbstractNouns +
  RuleRelativisticExpressions +
  RuleConfirmationExpressions +
  RuleRedundantExpressions +
  RuleTooLongExpressions +
  RuleAnaphoricReferences +
  RuleLiteraryStyle +
  RulePassive +
  RulePredSubjDistance +
  RulePredObjDistance +
  RuleInfVerbDistance +
  RuleMultiPartVerbs +
  RuleVerbalNouns +
  RuleDoubleComparison +
  RuleWrongValencyCase +
  RuleWrongVerbonominalCase +
  RuleIncompleteConjunction +
  sent_count +
```

```
  word_count +
  syllab_count +
  char_count +
  num_hapax

recipe_counts_base <- recipe(formula_counts, data = training_set)

recipe_counts_nocorr <- recipe_counts_base %>%
  step_normalize()
recipe_counts_nocorr
```

```
##
## -- Recipe ----------------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 34
##
## -- Operations
## * Centering and scaling for: <none>
```

```
recipe_counts <- recipe_counts_nocorr %>%
  add_corr_remove_step(training_set = training_set)
```

```
## # A tibble: 2 x 2
##   terms        id
##   <chr>        <chr>
## 1 syllab_count corr_BXbzh
## 2 char_count   corr_BXbzh
```

```
recipe_counts
```

```
##
## -- Recipe ----------------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 34
##
## -- Operations
## * Centering and scaling for: <none>
## * Correlation filter on: all_numeric_predictors()
```

Indicators, averages, and coefficients

```
formula_iac <- class ~
  RuleDoubleAdpos.max_allowable_distance +
  RuleDoubleAdpos.max_allowable_distance.v +
  RuleTooFewVerbs.min_verb_frac +
  RuleTooManyNegations.max_negation_frac +
  RuleTooManyNegations.max_negation_frac.v +
  RuleTooManyNegations.max_allowable_negations +
  RuleTooManyNegations.max_allowable_negations.v +
  RuleTooManyNominalConstructions.max_noun_frac +
  RuleTooManyNominalConstructions.max_noun_frac.v +
  RuleTooManyNominalConstructions.max_allowable_nouns +
  RuleTooManyNominalConstructions.max_allowable_nouns.v +
  RuleCaseRepetition.max_repetition_count +
  RuleCaseRepetition.max_repetition_count.v +
  RuleCaseRepetition.max_repetition_frac +
  RuleCaseRepetition.max_repetition_frac.v +
  RulePredSubjDistance.max_distance +
  RulePredSubjDistance.max_distance.v +
  RulePredObjDistance.max_distance +
  RulePredObjDistance.max_distance.v +
  RuleInfVerbDistance.max_distance +
  RuleInfVerbDistance.max_distance.v +
  RuleMultiPartVerbs.max_distance +
  RuleMultiPartVerbs.max_distance.v +
  RuleLongSentences.max_length +
  RuleLongSentences.max_length.v +
  RulePredAtClauseBeginning.max_order +
  RulePredAtClauseBeginning.max_order.v +
  cli +
  ari +
  entropy +
  ttr +
  mattr +
  mattr.v +
  maentropy +
  maentropy.v +
  mamr +
  verb_dist +
  activity +
  hpoint +
  atl +
  fre +
  fkgl +
  gf +
  smog

recipe_iac_base <- recipe(formula_iac, data = training_set)

recipe_iac_nocorr <- recipe_iac_base %>%
  step_normalize()
recipe_iac_nocorr
```

```
##
## -- Recipe ----------------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 44
##
## -- Operations
## * Centering and scaling for: <none>
```

```
recipe_iac <- recipe_iac_nocorr %>%
  add_corr_remove_step(training_set = training_set)
```

```
## # A tibble: 6 x 2
##   terms                                   id
##   <chr>                                   <chr>
## 1 RuleCaseRepetition.max_repetition_frac.v corr_2K3Tg
## 2 ari                                     corr_2K3Tg
## 3 maentropy                               corr_2K3Tg
## 4 atl                                     corr_2K3Tg
## 5 gf                                      corr_2K3Tg
## 6 smog                                    corr_2K3Tg
```

```
recipe_iac
```

```
##
## -- Recipe ----------------------------------------------------------------------
##
## -- Inputs
## Number of variables by role
## outcome:    1
## predictor: 44
##
## -- Operations
## * Centering and scaling for: <none>
## * Correlation filter on: all_numeric_predictors()
```

Null model

All variables

Remove correlating

```
train_null(recipe_all, folds)
```

```
## Null resamples:
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 4
##    splits           id     .metrics         .notes
##    <list>           <chr>  <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]>
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]>
## Null metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n  std_err .config
##   <chr>       <chr>      <dbl> <int>    <dbl> <chr>
## 1 accuracy    binary     0.564    10 0.00641  Preprocessor1_Model1
## 2 brier_class binary     0.246    10 0.000785 Preprocessor1_Model1
## 3 roc_auc     binary     0.5      10 0        Preprocessor1_Model1
```

```
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 4
##    splits           id     .metrics         .notes
##    <list>           <chr>  <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]>
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]>
```

Keep correlating

```
train_null(recipe_all_nocorr, folds)
```

```
## Null resamples:
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 4
##    splits           id     .metrics         .notes
##    <list>           <chr>  <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]>
```

```
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]>
## Null metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n  std_err .config
##   <chr>       <chr>      <dbl> <int>    <dbl> <chr>
## 1 accuracy    binary     0.564    10 0.00641  Preprocessor1_Model1
## 2 brier_class binary     0.246    10 0.000785 Preprocessor1_Model1
## 3 roc_auc     binary     0.5      10 0        Preprocessor1_Model1
```

```
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 4
##    splits           id     .metrics         .notes
##    <list>           <chr>  <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]>
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]>
```

Regular logistic regression

```
training_set_modif <- training_set %>%
  mutate(across(class, ~ .x == "good")) %>%
  mutate(across(RuleAbstractNouns:word_count, ~ scale(.x)))
```

All variables

```
glm(
  formula_all,
  data = training_set_modif,
  family = binomial(link = "logit")
) %>% summary()
```

```
##
## Call:
## glm(formula = formula_all, family = binomial(link = "logit"),
##     data = training_set_modif)
##
## Coefficients: (1 not defined because of singularities)
##                                                  Estimate Std. Error
## (Intercept)                                     -0.690955   0.168762
## RuleGPcoordovs                                  -0.131060   0.123109
```

```
## RuleGPdeverbaddr                                    -0.117172   0.136160
## RuleGPpatinstr                                      -0.108238   0.131956
## RuleGPdeverbsubj                                    -0.136787   0.114819
## RuleGPadjective                                      0.310408   0.200549
## RuleGPpatbenperson                                  -0.100164   0.133302
## RuleGPwordorder                                     -0.106513   0.150526
## RuleDoubleAdpos                                     -0.124214   0.158907
## RuleDoubleAdpos.max_allowable_distance               0.012872   0.272618
## RuleDoubleAdpos.max_allowable_distance.v             0.064192   0.221548
## RuleAmbiguousRegards                                 0.348388   0.230223
## RuleReflexivePassWithAnimSubj                       -0.028087   0.145173
## RuleTooFewVerbs.min_verb_frac                       -1.712632   0.529857
## RuleTooManyNegations.max_negation_frac               0.033468   0.205385
## RuleTooManyNegations.max_negation_frac.v             0.105700   0.163240
## RuleTooManyNegations.max_allowable_negations         0.188081   0.256184
## RuleTooManyNegations.max_allowable_negations.v      -0.129654   0.226260
## RuleTooManyNominalConstructions.max_noun_frac       -0.267628   0.228974
## RuleTooManyNominalConstructions.max_noun_frac.v      0.096042   0.157962
## RuleTooManyNominalConstructions.max_allowable_nouns  0.490385   0.501294
## RuleFunctionWordRepetition                          -0.326041   0.237704
## RuleCaseRepetition.max_repetition_count              0.011173   0.379015
## RuleCaseRepetition.max_repetition_count.v           -0.338944   0.195320
## RuleCaseRepetition.max_repetition_frac               1.362738   1.037439
## RuleCaseRepetition.max_repetition_frac.v            1.701358   1.016826
## RuleWeakMeaningWords                                -0.081026   0.138318
## RuleAbstractNouns                                    0.114087   0.134712
## RuleRelativisticExpressions                         -0.258514   0.147301
## RuleConfirmationExpressions                          0.202065   0.151704
## RuleRedundantExpressions                             0.052902   0.162831
## RuleTooLongExpressions                               0.280119   0.154476
## RuleAnaphoricReferences                              0.579313   0.161739
## RuleLiteraryStyle                                   -0.333322   0.152971
## RulePassive                                         -0.331986   0.203950
## RulePredSubjDistance                                 0.478727   0.229624
## RulePredSubjDistance.max_distance                   -0.453011   0.294763
## RulePredSubjDistance.max_distance.v                 -0.061425   0.208988
## RulePredObjDistance                                  0.030800   0.256329
## RulePredObjDistance.max_distance                    -0.507437   0.312116
## RulePredObjDistance.max_distance.v                   0.061851   0.188965
## RuleInfVerbDistance                                  0.267386   0.259383
## RuleInfVerbDistance.max_distance                     0.286577   0.142475
## RuleInfVerbDistance.max_distance.v                  -0.295289   0.174780
## RuleMultiPartVerbs                                   0.481640   0.250046
## RuleMultiPartVerbs.max_distance                      0.065185   0.234704
## RuleMultiPartVerbs.max_distance.v                    0.251675   0.205316
## RuleLongSentences.max_length                         3.056180   0.983675
## RuleLongSentences.max_length.v                       0.743749   0.240055
## RulePredAtClauseBeginning.max_order                 -0.230326   0.363884
## RulePredAtClauseBeginning.max_order.v                0.006383   0.248539
## RuleVerbalNouns                                     -0.109267   0.157207
## RuleDoubleComparison                                -0.033847   0.115238
## RuleWrongValencyCase                                 0.112750   0.123573
## RuleWrongVerbonominalCase                            0.101861   0.151825
## RuleIncompleteConjunction                           -0.232348   0.135531
```

```
## sent_count                                            0.943246   0.711149
## word_count                                           -4.925570   3.905163
## syllab_count                                        -11.660720   6.284953
## char_count                                           16.679088   8.301499
## cli                                                  -2.087343   2.190863
## ari                                                  -5.021019   1.910463
## num_hapax                                            -0.214638   1.007843
## entropy                                              -0.493658   0.379718
## ttr                                                   0.085436   1.331619
## mattr                                                -1.103050   1.099941
## mattr.v                                              -0.570758   0.472312
## maentropy                                             0.849252   1.130609
## maentropy.v                                           1.478828   0.791809
## mamr                                                 -0.064783   0.293329
## verb_dist                                             0.287567   0.325602
## activity                                              1.700625   0.561043
## hpoint                                               -1.284726   0.865501
## atl                                                   2.316932   2.535645
## fre                                                  -2.531607   1.053731
## fkgl                                                        NA         NA
## gf                                                   -2.156934   2.445513
## smog                                                  1.621313   1.997701
##                                                     z value Pr(>|z|)
## (Intercept)                                          -4.094 4.24e-05 ***
## RuleGPcoordovs                                       -1.065 0.287064
## RuleGPdeverbaddr                                     -0.861 0.389490
## RuleGPpatinstr                                       -0.820 0.412070
## RuleGPdeverbsubj                                     -1.191 0.233526
## RuleGPadjective                                       1.548 0.121673
## RuleGPpatbenperson                                   -0.751 0.452405
## RuleGPwordorder                                      -0.708 0.479188
## RuleDoubleAdpos                                      -0.782 0.434405
## RuleDoubleAdpos.max_allowable_distance                0.047 0.962340
## RuleDoubleAdpos.max_allowable_distance.v              0.290 0.772015
## RuleAmbiguousRegards                                  1.513 0.130213
## RuleReflexivePassWithAnimSubj                        -0.193 0.846589
## RuleTooFewVerbs.min_verb_frac                        -3.232 0.001228 **
## RuleTooManyNegations.max_negation_frac                0.163 0.870556
## RuleTooManyNegations.max_negation_frac.v              0.648 0.517299
## RuleTooManyNegations.max_allowable_negations          0.734 0.462849
## RuleTooManyNegations.max_allowable_negations.v       -0.573 0.566625
## RuleTooManyNominalConstructions.max_noun_frac        -1.169 0.242478
## RuleTooManyNominalConstructions.max_noun_frac.v       0.608 0.543181
## RuleTooManyNominalConstructions.max_allowable_nouns   0.978 0.327957
## RuleFunctionWordRepetition                           -1.372 0.170181
## RuleCaseRepetition.max_repetition_count               0.029 0.976483
## RuleCaseRepetition.max_repetition_count.v            -1.735 0.082683 .
## RuleCaseRepetition.max_repetition_frac                1.314 0.188994
## RuleCaseRepetition.max_repetition_frac.v              1.673 0.094287 .
## RuleWeakMeaningWords                                 -0.586 0.558014
## RuleAbstractNouns                                     0.847 0.397052
## RuleRelativisticExpressions                          -1.755 0.079259 .
## RuleConfirmationExpressions                           1.332 0.182871
## RuleRedundantExpressions                              0.325 0.745268
```

```
## RuleTooLongExpressions                                1.813 0.069778 .
## RuleAnaphoricReferences                               3.582 0.000341 ***
## RuleLiteraryStyle                                    -2.179 0.029333 *
## RulePassive                                          -1.628 0.103571
## RulePredSubjDistance                                  2.085 0.037084 *
## RulePredSubjDistance.max_distance                    -1.537 0.124327
## RulePredSubjDistance.max_distance.v                  -0.294 0.768824
## RulePredObjDistance                                   0.120 0.904356
## RulePredObjDistance.max_distance                     -1.626 0.103993
## RulePredObjDistance.max_distance.v                    0.327 0.743428
## RuleInfVerbDistance                                   1.031 0.302609
## RuleInfVerbDistance.max_distance                      2.011 0.044280 *
## RuleInfVerbDistance.max_distance.v                   -1.689 0.091126 .
## RuleMultiPartVerbs                                    1.926 0.054079 .
## RuleMultiPartVerbs.max_distance                       0.278 0.781217
## RuleMultiPartVerbs.max_distance.v                     1.226 0.220276
## RuleLongSentences.max_length                          3.107 0.001891 **
## RuleLongSentences.max_length.v                        3.098 0.001947 **
## RulePredAtClauseBeginning.max_order                  -0.633 0.526757
## RulePredAtClauseBeginning.max_order.v                 0.026 0.979512
## RuleVerbalNouns                                      -0.695 0.487021
## RuleDoubleComparison                                 -0.294 0.768975
## RuleWrongValencyCase                                  0.912 0.361548
## RuleWrongVerbonominalCase                             0.671 0.502277
## RuleIncompleteConjunction                            -1.714 0.086464 .
## sent_count                                            1.326 0.184718
## word_count                                           -1.261 0.207202
## syllab_count                                         -1.855 0.063548 .
## char_count                                            2.009 0.044520 *
## cli                                                  -0.953 0.340717
## ari                                                  -2.628 0.008585 **
## num_hapax                                            -0.213 0.831352
## entropy                                              -1.300 0.193578
## ttr                                                   0.064 0.948843
## mattr                                                -1.003 0.315945
## mattr.v                                              -1.208 0.226880
## maentropy                                             0.751 0.452565
## maentropy.v                                           1.868 0.061810 .
## mamr                                                 -0.221 0.825206
## verb_dist                                             0.883 0.377136
## activity                                              3.031 0.002436 **
## hpoint                                               -1.484 0.137710
## atl                                                   0.914 0.360851
## fre                                                  -2.403 0.016283 *
## fkgl                                                     NA       NA
## gf                                                   -0.882 0.377779
## smog                                                  0.812 0.417027
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 842.39  on 614  degrees of freedom
## Residual deviance: 435.23  on 538  degrees of freedom
```

```
## AIC: 589.23
##
## Number of Fisher Scoring iterations: 6
```

Indicators, averages, and coefficients

```
glm(
  formula_iac,
  data = training_set_modif,
  family = binomial(link = "logit")
) %>% summary()
```

```
##
## Call:
## glm(formula = formula_iac, family = binomial(link = "logit"),
##     data = training_set_modif)
##
## Coefficients: (1 not defined because of singularities)
##                                                    Estimate Std. Error
## (Intercept)                                      -0.5302579  0.1338761
## RuleDoubleAdpos.max_allowable_distance            0.1540172  0.1956006
## RuleDoubleAdpos.max_allowable_distance.v         -0.1760912  0.1695237
## RuleTooFewVerbs.min_verb_frac                    -1.4110093  0.4176640
## RuleTooManyNegations.max_negation_frac           -0.0773115  0.1729624
## RuleTooManyNegations.max_negation_frac.v          0.1307729  0.1313145
## RuleTooManyNegations.max_allowable_negations      0.1195366  0.2276293
## RuleTooManyNegations.max_allowable_negations.v   -0.2237944  0.1917024
## RuleTooManyNominalConstructions.max_noun_frac    -0.3183450  0.1846450
## RuleTooManyNominalConstructions.max_noun_frac.v   0.0844108  0.1348965
## RuleTooManyNominalConstructions.max_allowable_nouns  0.2398039  0.4014291
## RuleTooManyNominalConstructions.max_allowable_nouns.v -0.2243448  0.1858581
## RuleCaseRepetition.max_repetition_count           0.0153097  0.2936628
## RuleCaseRepetition.max_repetition_count.v        -0.3720261  0.1683255
## RuleCaseRepetition.max_repetition_frac            0.6504295  0.8036846
## RuleCaseRepetition.max_repetition_frac.v          0.9997573  0.7835061
## RulePredSubjDistance.max_distance                -0.5140230  0.2803484
## RulePredSubjDistance.max_distance.v              -0.0003185  0.1779153
## RulePredObjDistance.max_distance                 -0.3092264  0.2519789
## RulePredObjDistance.max_distance.v                0.0036661  0.1619852
## RuleInfVerbDistance.max_distance                  0.1830175  0.1150805
## RuleInfVerbDistance.max_distance.v               -0.3463788  0.1457475
## RuleMultiPartVerbs.max_distance                   0.1725754  0.1989849
## RuleMultiPartVerbs.max_distance.v                 0.1675762  0.1776904
## RuleLongSentences.max_length                      3.0083994  0.9077630
## RuleLongSentences.max_length.v                    0.6661785  0.1846859
## RulePredAtClauseBeginning.max_order               0.0594934  0.4032334
## RulePredAtClauseBeginning.max_order.v            -0.1710516  0.2117331
## cli                                              -1.2777424  1.7019639
## ari                                              -4.1365511  1.3083541
## entropy                                           0.0399705  0.3004075
## ttr                                              -0.2438371  0.3227898
## mattr                                            -1.0034306  0.8635519
## mattr.v                                          -0.5590522  0.4005484
## maentropy                                         0.6830763  0.8769725
```

```
## maentropy.v                                            1.1017018  0.6383895
## mamr                                                   0.0922477  0.2305422
## verb_dist                                              0.4195557  0.2597051
## activity                                               1.8867365  0.3899413
## hpoint                                                -0.4416800  0.3623207
## atl                                                    2.1659999  1.8484154
## fre                                                   -1.9400652  0.5324702
## fkgl                                                         NA         NA
## gf                                                    -1.5127110  2.0926875
## smog                                                   0.5665066  1.6903358
##                                                       z value Pr(>|z|)
## (Intercept)                                            -3.961 7.47e-05 ***
## RuleDoubleAdpos.max_allowable_distance                  0.787 0.431044
## RuleDoubleAdpos.max_allowable_distance.v               -1.039 0.298925
## RuleTooFewVerbs.min_verb_frac                          -3.378 0.000729 ***
## RuleTooManyNegations.max_negation_frac                 -0.447 0.654886
## RuleTooManyNegations.max_negation_frac.v                0.996 0.319311
## RuleTooManyNegations.max_allowable_negations            0.525 0.599488
## RuleTooManyNegations.max_allowable_negations.v         -1.167 0.243047
## RuleTooManyNominalConstructions.max_noun_frac          -1.724 0.084691 .
## RuleTooManyNominalConstructions.max_noun_frac.v         0.626 0.531482
## RuleTooManyNominalConstructions.max_allowable_nouns     0.597 0.550257
## RuleTooManyNominalConstructions.max_allowable_nouns.v  -1.207 0.227403
## RuleCaseRepetition.max_repetition_count                 0.052 0.958422
## RuleCaseRepetition.max_repetition_count.v              -2.210 0.027094 *
## RuleCaseRepetition.max_repetition_frac                  0.809 0.418337
## RuleCaseRepetition.max_repetition_frac.v                1.276 0.201954
## RulePredSubjDistance.max_distance                      -1.834 0.066726 .
## RulePredSubjDistance.max_distance.v                    -0.002 0.998572
## RulePredObjDistance.max_distance                       -1.227 0.219751
## RulePredObjDistance.max_distance.v                      0.023 0.981944
## RuleInfVerbDistance.max_distance                        1.590 0.111757
## RuleInfVerbDistance.max_distance.v                     -2.377 0.017475 *
## RuleMultiPartVerbs.max_distance                         0.867 0.385789
## RuleMultiPartVerbs.max_distance.v                       0.943 0.345640
## RuleLongSentences.max_length                            3.314 0.000919 ***
## RuleLongSentences.max_length.v                          3.607 0.000310 ***
## RulePredAtClauseBeginning.max_order                     0.148 0.882705
## RulePredAtClauseBeginning.max_order.v                  -0.808 0.419169
## cli                                                    -0.751 0.452806
## ari                                                    -3.162 0.001569 **
## entropy                                                 0.133 0.894151
## ttr                                                    -0.755 0.450006
## mattr                                                  -1.162 0.245243
## mattr.v                                                -1.396 0.162800
## maentropy                                               0.779 0.436037
## maentropy.v                                             1.726 0.084392 .
## mamr                                                    0.400 0.689058
## verb_dist                                               1.616 0.106201
## activity                                                4.839 1.31e-06 ***
## hpoint                                                 -1.219 0.222833
## atl                                                     1.172 0.241272
## fre                                                    -3.644 0.000269 ***
## fkgl                                                       NA       NA
```

```
## gf                                                    -0.723 0.469769
## smog                                                   0.335 0.737516
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 842.39  on 614  degrees of freedom
## Residual deviance: 515.48  on 571  degrees of freedom
## AIC: 603.48
##
## Number of Fisher Scoring iterations: 6
```

Counts

```
glm(
  formula_counts,
  data = training_set_modif,
  family = binomial(link = "logit")
) %>% summary()
```

```
##
## Call:
## glm(formula = formula_counts, family = binomial(link = "logit"),
##     data = training_set_modif)
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -0.56069    0.12447  -4.504 6.65e-06 ***
## RuleGPcoordovs                 -0.09465    0.09994  -0.947 0.343599
## RuleGPdeverbaddr               -0.13951    0.11239  -1.241 0.214479
## RuleGPpatinstr                 -0.01627    0.10233  -0.159 0.873642
## RuleGPdeverbsubj               -0.15347    0.13130  -1.169 0.242451
## RuleGPadjective                 0.20509    0.16118   1.272 0.203236
## RuleGPpatbenperson             -0.01710    0.09478  -0.180 0.856849
## RuleGPwordorder                -0.13622    0.11871  -1.148 0.251151
## RuleDoubleAdpos                -0.19946    0.11189  -1.783 0.074653 .
## RuleAmbiguousRegards            0.27562    0.18738   1.471 0.141327
## RuleReflexivePassWithAnimSubj   0.04515    0.11264   0.401 0.688563
## RuleFunctionWordRepetition     -0.01340    0.09365  -0.143 0.886253
## RuleWeakMeaningWords           -0.01933    0.11109  -0.174 0.861878
## RuleAbstractNouns               0.02509    0.11184   0.224 0.822505
## RuleRelativisticExpressions    -0.24953    0.13760  -1.813 0.069774 .
## RuleConfirmationExpressions     0.12340    0.12419   0.994 0.320418
## RuleRedundantExpressions       -0.01600    0.13709  -0.117 0.907063
## RuleTooLongExpressions          0.30589    0.11457   2.670 0.007589 **
## RuleAnaphoricReferences         0.43908    0.12733   3.448 0.000564 ***
## RuleLiteraryStyle              -0.46045    0.12574  -3.662 0.000250 ***
## RulePassive                    -0.48705    0.14361  -3.391 0.000695 ***
## RulePredSubjDistance            0.23709    0.14857   1.596 0.110543
## RulePredObjDistance             0.15677    0.14881   1.054 0.292106
## RuleInfVerbDistance             0.11508    0.14855   0.775 0.438500
## RuleMultiPartVerbs              0.34643    0.15343   2.258 0.023948 *
## RuleVerbalNouns                 0.07022    0.12034   0.583 0.559584
```

```
## RuleDoubleComparison          -0.02175    0.10157  -0.214 0.830433
## RuleWrongValencyCase           0.05330    0.11167   0.477 0.633146
## RuleWrongVerbonominalCase      0.10638    0.11795   0.902 0.367090
## RuleIncompleteConjunction     -0.15799    0.13062  -1.210 0.226443
## sent_count                     1.54054    0.42541   3.621 0.000293 ***
## word_count                    -3.82029    1.87402  -2.039 0.041494 *
## syllab_count                  -1.21636    3.27351  -0.372 0.710209
## char_count                     2.86370    3.88848   0.736 0.461452
## num_hapax                     -0.11331    0.17307  -0.655 0.512657
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 842.39  on 614  degrees of freedom
## Residual deviance: 540.38  on 580  degrees of freedom
## AIC: 610.38
##
## Number of Fisher Scoring iterations: 6
```

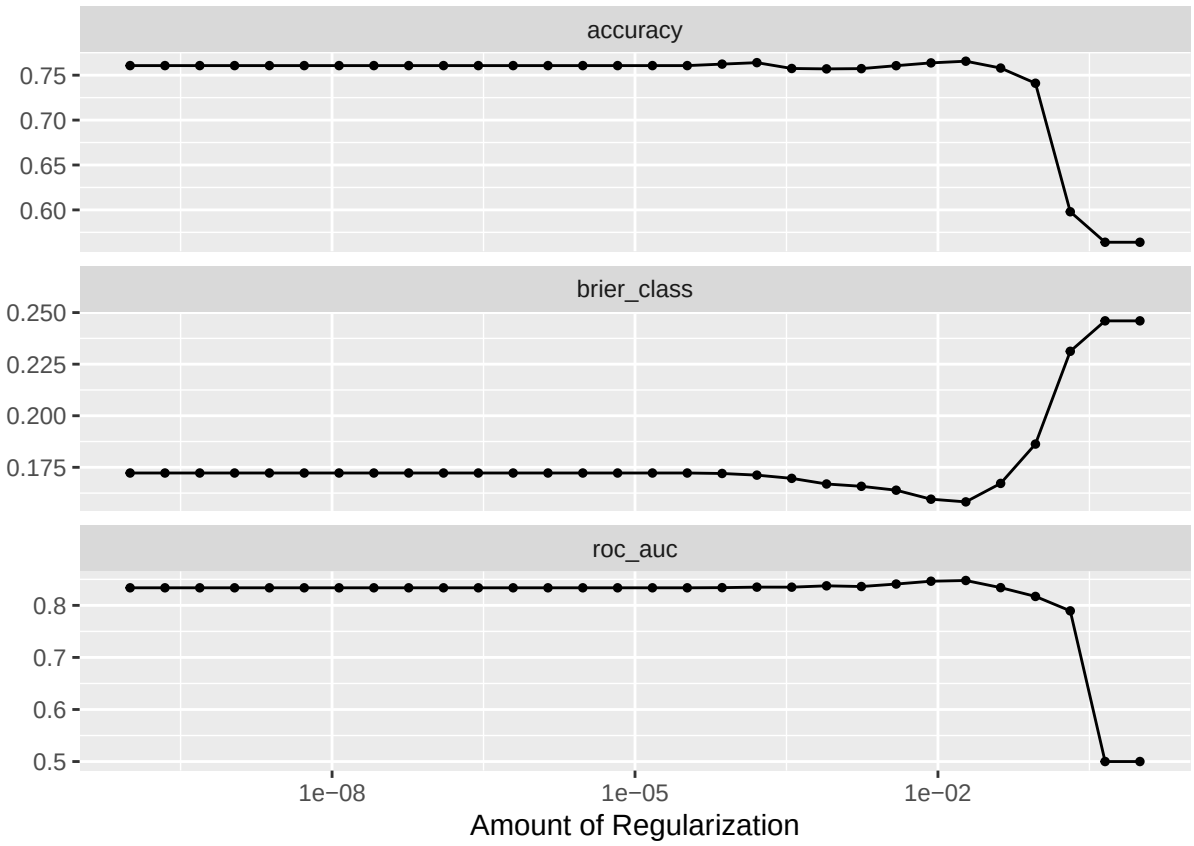
Lasso

All variables

Remove correlating

```
train_lasso(recipe_all, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##    penalty .metric .estimator  mean     n std_err .config
##      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.0189   roc_auc binary     0.848    10  0.0110 Preprocessor1_Model25
## 2 0.00853  roc_auc binary     0.846    10  0.0118 Preprocessor1_Model24
## 3 0.00386  roc_auc binary     0.841    10  0.0118 Preprocessor1_Model23
## 4 0.000788 roc_auc binary     0.838    10  0.0116 Preprocessor1_Model21
## 5 0.00174  roc_auc binary     0.836    10  0.0117 Preprocessor1_Model22
## # A tibble: 5 x 7
##        penalty .metric  .estimator  mean     n std_err .config
##          <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.0189       accuracy binary     0.766    10 0.0112  Preprocessor1_Model25
## 2 0.000161     accuracy binary     0.764    10 0.0108  Preprocessor1_Model19
## 3 0.00853      accuracy binary     0.764    10 0.0110  Preprocessor1_Model24
## 4 0.0000728    accuracy binary     0.762    10 0.00998 Preprocessor1_Model18
## 5 0.0000000001 accuracy binary     0.761    10 0.0103  Preprocessor1_Model01
## Best accuracy:
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <chr>
## 1  0.0418 Preprocessor1_Model26
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.0417531893656041
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 68 x 3
##    term                                              estimate penalty
##    <chr>                                                <dbl>   <dbl>
##  1 RuleLiteraryStyle                                  -0.357   0.0418
##  2 (Intercept)                                        -0.322   0.0418
##  3 entropy                                            -0.188   0.0418
##  4 fkgl                                               -0.135   0.0418
##  5 mattr                                              -0.0678  0.0418
##  6 RulePassive                                        -0.0448  0.0418
##  7 RuleTooManyNegations.max_allowable_negations       -0.0297  0.0418
##  8 RuleGPcoordovs                                      0       0.0418
##  9 RuleGPdeverbaddr                                    0       0.0418
## 10 RuleGPpatinstr                                      0       0.0418
## 11 RuleGPdeverbsubj                                    0       0.0418
## 12 RuleGPadjective                                     0       0.0418
## 13 RuleGPpatbenperson                                  0       0.0418
## 14 RuleGPwordorder                                     0       0.0418
## 15 RuleDoubleAdpos                                     0       0.0418
## 16 RuleDoubleAdpos.max_allowable_distance              0       0.0418
## 17 RuleDoubleAdpos.max_allowable_distance.v            0       0.0418
## 18 RuleAmbiguousRegards                                0       0.0418
## 19 RuleReflexivePassWithAnimSubj                       0       0.0418
## 20 RuleTooFewVerbs.min_verb_frac                       0       0.0418
## 21 RuleTooManyNegations.max_negation_frac              0       0.0418
## 22 RuleTooManyNegations.max_negation_frac.v            0       0.0418
## 23 RuleTooManyNegations.max_allowable_negations.v      0       0.0418
## 24 RuleTooManyNominalConstructions.max_noun_frac       0       0.0418
## 25 RuleTooManyNominalConstructions.max_noun_frac.v     0       0.0418
## 26 RuleTooManyNominalConstructions.max_allowable_nouns 0       0.0418
## 27 RuleFunctionWordRepetition                          0       0.0418
## 28 RuleCaseRepetition.max_repetition_count             0       0.0418
## 29 RuleCaseRepetition.max_repetition_count.v           0       0.0418
## 30 RuleCaseRepetition.max_repetition_frac              0       0.0418
## 31 RuleWeakMeaningWords                                0       0.0418
## 32 RuleAbstractNouns                                   0       0.0418
## 33 RuleRelativisticExpressions                         0       0.0418
## 34 RuleConfirmationExpressions                         0       0.0418
## 35 RuleRedundantExpressions                            0       0.0418
## 36 RuleTooLongExpressions                              0       0.0418
```

```
## 37 RulePredSubjDistance                                  0       0.0418
## 38 RulePredSubjDistance.max_distance                     0       0.0418
## 39 RulePredSubjDistance.max_distance.v                   0       0.0418
## 40 RulePredObjDistance                                   0       0.0418
## 41 RulePredObjDistance.max_distance                      0       0.0418
## 42 RulePredObjDistance.max_distance.v                    0       0.0418
## 43 RuleInfVerbDistance                                   0       0.0418
## 44 RuleInfVerbDistance.max_distance                      0       0.0418
## 45 RuleInfVerbDistance.max_distance.v                    0       0.0418
## 46 RuleMultiPartVerbs.max_distance                       0       0.0418
## 47 RuleMultiPartVerbs.max_distance.v                     0       0.0418
## 48 RuleLongSentences.max_length                          0       0.0418
## 49 RulePredAtClauseBeginning.max_order                   0       0.0418
## 50 RulePredAtClauseBeginning.max_order.v                 0       0.0418
## 51 RuleVerbalNouns                                       0       0.0418
## 52 RuleDoubleComparison                                  0       0.0418
## 53 RuleWrongValencyCase                                  0       0.0418
## 54 RuleWrongVerbonominalCase                             0       0.0418
## 55 RuleIncompleteConjunction                             0       0.0418
## 56 sent_count                                            0       0.0418
## 57 syllab_count                                          0       0.0418
## 58 num_hapax                                             0       0.0418
## 59 mattr.v                                               0       0.0418
## 60 maentropy.v                                           0       0.0418
## 61 verb_dist                                             0       0.0418
## 62 fre                                                   0       0.0418
## 63 RuleLongSentences.max_length.v                        0.0227  0.0418
## 64 RuleMultiPartVerbs                                    0.0767  0.0418
## 65 mamr                                                  0.0830  0.0418
## 66 RuleAnaphoricReferences                               0.0940  0.0418
## 67 cli                                                   0.189   0.0418
## 68 activity                                              0.605   0.0418
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.238700
## 2   1  2.87 0.217500
## 3   1  5.27 0.198200
## 4   1  7.28 0.180600
## 5   1  8.99 0.164600
## 6   4 10.65 0.149900
```

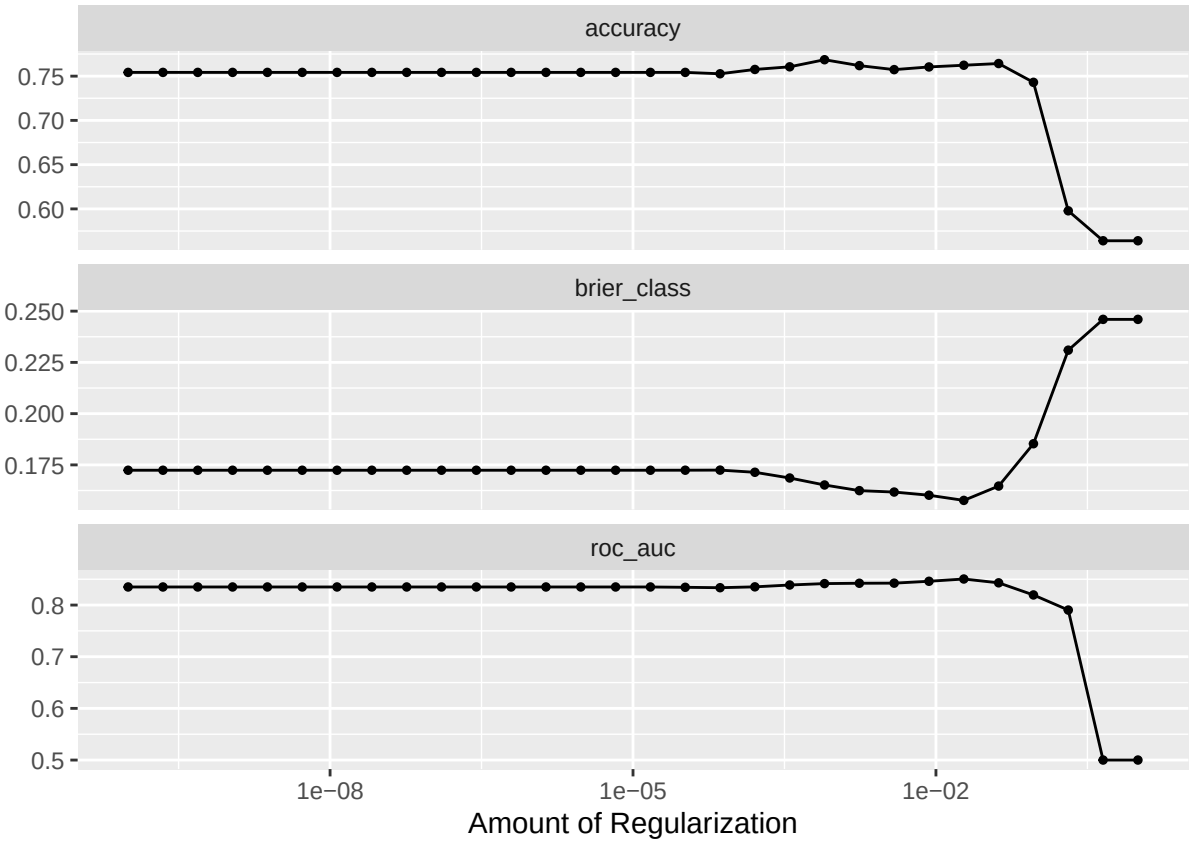
```
## 7   5 12.85 0.136600
## 8   5 14.75 0.124500
## 9   5 16.39 0.113400
## 10  4 17.79 0.103300
## 11  4 18.99 0.094160
## 12  5 20.04 0.085800
## 13  5 21.14 0.078170
## 14  7 22.23 0.071230
## 15  7 23.35 0.064900
## 16  8 24.35 0.059140
## 17 10 25.42 0.053880
## 18 11 26.52 0.049100
## 19 12 27.49 0.044730
## 20 12 28.55 0.040760
## 21 14 29.53 0.037140
## 22 15 30.47 0.033840
## 23 17 31.33 0.030830
## 24 20 32.25 0.028090
## 25 20 33.10 0.025600
## 26 22 34.04 0.023320
## 27 22 34.86 0.021250
## 28 25 35.58 0.019360
## 29 27 36.26 0.017640
## 30 29 36.89 0.016080
## 31 29 37.44 0.014650
## 32 31 37.94 0.013350
## 33 34 38.42 0.012160
## 34 35 38.94 0.011080
## 35 37 39.44 0.010100
## 36 38 39.91 0.009200
## 37 41 40.34 0.008382
## 38 44 40.76 0.007638
## 39 45 41.14 0.006959
## 40 45 41.46 0.006341
## 41 46 41.74 0.005778
## 42 47 42.04 0.005264
## 43 49 42.34 0.004797
## 44 49 42.64 0.004371
## 45 52 42.89 0.003982
## 46 52 43.14 0.003629
##
## ...
## and 46 more lines.
```

Keep correlating

```
train_lasso(recipe_all_nocorr, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##   penalty .metric .estimator  mean     n std_err .config
##     <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.0189  roc_auc binary     0.850    10 0.0119  Preprocessor1_Model25
## 2 0.00853 roc_auc binary     0.846    10 0.0118  Preprocessor1_Model24
## 3 0.0418  roc_auc binary     0.843    10 0.00947 Preprocessor1_Model26
## 4 0.00386 roc_auc binary     0.842    10 0.0122  Preprocessor1_Model23
## 5 0.00174 roc_auc binary     0.842    10 0.0118  Preprocessor1_Model22
## # A tibble: 5 x 7
##    penalty .metric  .estimator  mean     n std_err .config
##      <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.000788 accuracy binary     0.769    10 0.00978 Preprocessor1_Model21
```

```
## 2 0.0418   accuracy binary     0.764    10 0.0136  Preprocessor1_Model26
## 3 0.0189   accuracy binary     0.762    10 0.0124  Preprocessor1_Model25
## 4 0.00174  accuracy binary     0.762    10 0.0124  Preprocessor1_Model22
## 5 0.000356 accuracy binary     0.761    10 0.00991 Preprocessor1_Model20
## Best accuracy:
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <chr>
## 1  0.0418 Preprocessor1_Model26
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.0417531893656041
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 78 x 3
##    term                                                estimate penalty
##    <chr>                                                  <dbl>   <dbl>
##  1 (Intercept)                                          -0.333    0.0418
##  2 RuleLiteraryStyle                                    -0.329    0.0418
##  3 smog                                                 -0.222    0.0418
##  4 entropy                                              -0.189    0.0418
##  5 maentropy                                            -0.109    0.0418
##  6 RulePassive                                          -0.00237  0.0418
##  7 RuleGPcoordovs                                        0        0.0418
##  8 RuleGPdeverbaddr                                      0        0.0418
##  9 RuleGPpatinstr                                        0        0.0418
## 10 RuleGPdeverbsubj                                      0        0.0418
## 11 RuleGPadjective                                       0        0.0418
## 12 RuleGPpatbenperson                                    0        0.0418
## 13 RuleGPwordorder                                       0        0.0418
## 14 RuleDoubleAdpos                                       0        0.0418
## 15 RuleDoubleAdpos.max_allowable_distance                0        0.0418
## 16 RuleDoubleAdpos.max_allowable_distance.v              0        0.0418
## 17 RuleAmbiguousRegards                                  0        0.0418
## 18 RuleReflexivePassWithAnimSubj                         0        0.0418
## 19 RuleTooFewVerbs.min_verb_frac                         0        0.0418
## 20 RuleTooManyNegations.max_negation_frac                0        0.0418
## 21 RuleTooManyNegations.max_negation_frac.v              0        0.0418
## 22 RuleTooManyNegations.max_allowable_negations          0        0.0418
```

```
## 23 RuleTooManyNegations.max_allowable_negations.v       0        0.0418
## 24 RuleTooManyNominalConstructions.max_noun_frac        0        0.0418
## 25 RuleTooManyNominalConstructions.max_noun_frac.v      0        0.0418
## 26 RuleTooManyNominalConstructions.max_allowable_nouns  0        0.0418
## 27 RuleFunctionWordRepetition                           0        0.0418
## 28 RuleCaseRepetition.max_repetition_count              0        0.0418
## 29 RuleCaseRepetition.max_repetition_count.v            0        0.0418
## 30 RuleCaseRepetition.max_repetition_frac               0        0.0418
## 31 RuleCaseRepetition.max_repetition_frac.v             0        0.0418
## 32 RuleWeakMeaningWords                                 0        0.0418
## 33 RuleAbstractNouns                                    0        0.0418
## 34 RuleRelativisticExpressions                          0        0.0418
## 35 RuleConfirmationExpressions                          0        0.0418
## 36 RuleRedundantExpressions                             0        0.0418
## 37 RuleTooLongExpressions                               0        0.0418
## 38 RulePredSubjDistance                                 0        0.0418
## 39 RulePredSubjDistance.max_distance                    0        0.0418
## 40 RulePredSubjDistance.max_distance.v                  0        0.0418
## 41 RulePredObjDistance                                  0        0.0418
## 42 RulePredObjDistance.max_distance                     0        0.0418
## 43 RulePredObjDistance.max_distance.v                   0        0.0418
## 44 RuleInfVerbDistance                                  0        0.0418
## 45 RuleInfVerbDistance.max_distance                     0        0.0418
## 46 RuleInfVerbDistance.max_distance.v                   0        0.0418
## 47 RuleMultiPartVerbs.max_distance                      0        0.0418
## 48 RuleMultiPartVerbs.max_distance.v                    0        0.0418
## 49 RuleLongSentences.max_length                         0        0.0418
## 50 RulePredAtClauseBeginning.max_order                  0        0.0418
## 51 RulePredAtClauseBeginning.max_order.v                0        0.0418
## 52 RuleVerbalNouns                                      0        0.0418
## 53 RuleDoubleComparison                                 0        0.0418
## 54 RuleWrongValencyCase                                 0        0.0418
## 55 RuleWrongVerbonominalCase                            0        0.0418
## 56 RuleIncompleteConjunction                            0        0.0418
## 57 sent_count                                           0        0.0418
## 58 word_count                                           0        0.0418
## 59 syllab_count                                         0        0.0418
## 60 char_count                                           0        0.0418
## 61 cli                                                  0        0.0418
## 62 ari                                                  0        0.0418
## 63 num_hapax                                            0        0.0418
## 64 ttr                                                  0        0.0418
## 65 mattr                                                0        0.0418
## 66 mattr.v                                              0        0.0418
## 67 maentropy.v                                          0        0.0418
## 68 mamr                                                 0        0.0418
## 69 verb_dist                                            0        0.0418
## 70 hpoint                                               0        0.0418
## 71 fre                                                  0        0.0418
## 72 fkgl                                                 0        0.0418
## 73 gf                                                   0        0.0418
## 74 RuleMultiPartVerbs                                   0.00595  0.0418
## 75 RuleLongSentences.max_length.v                       0.0208   0.0418
## 76 RuleAnaphoricReferences                              0.106    0.0418
```

```
## 77 atl                                                  0.326    0.0418
## 78 activity                                              0.559    0.0418
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model ---------------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.238700
## 2   1  2.87 0.217500
## 3   3  5.53 0.198200
## 4   3  7.92 0.180600
## 5   3  9.94 0.164600
## 6   3 11.67 0.149900
## 7   4 13.31 0.136600
## 8   5 15.15 0.124500
## 9   6 16.79 0.113400
## 10  6 18.34 0.103300
## 11  6 19.68 0.094160
## 12  6 20.96 0.085800
## 13  6 22.27 0.078170
## 14  7 23.45 0.071230
## 15  7 24.49 0.064900
## 16  8 25.41 0.059140
## 17  8 26.46 0.053880
## 18  8 27.37 0.049100
## 19  7 28.13 0.044730
## 20 10 29.04 0.040760
## 21 12 30.00 0.037140
## 22 13 30.87 0.033840
## 23 16 31.68 0.030830
## 24 20 32.62 0.028090
## 25 20 33.44 0.025600
## 26 23 34.33 0.023320
## 27 23 35.13 0.021250
## 28 25 35.82 0.019360
## 29 25 36.49 0.017640
## 30 27 37.06 0.016080
## 31 28 37.59 0.014650
## 32 30 38.07 0.013350
## 33 29 38.49 0.012160
## 34 33 38.94 0.011080
## 35 38 39.45 0.010100
## 36 42 40.00 0.009200
## 37 44 40.51 0.008382
```

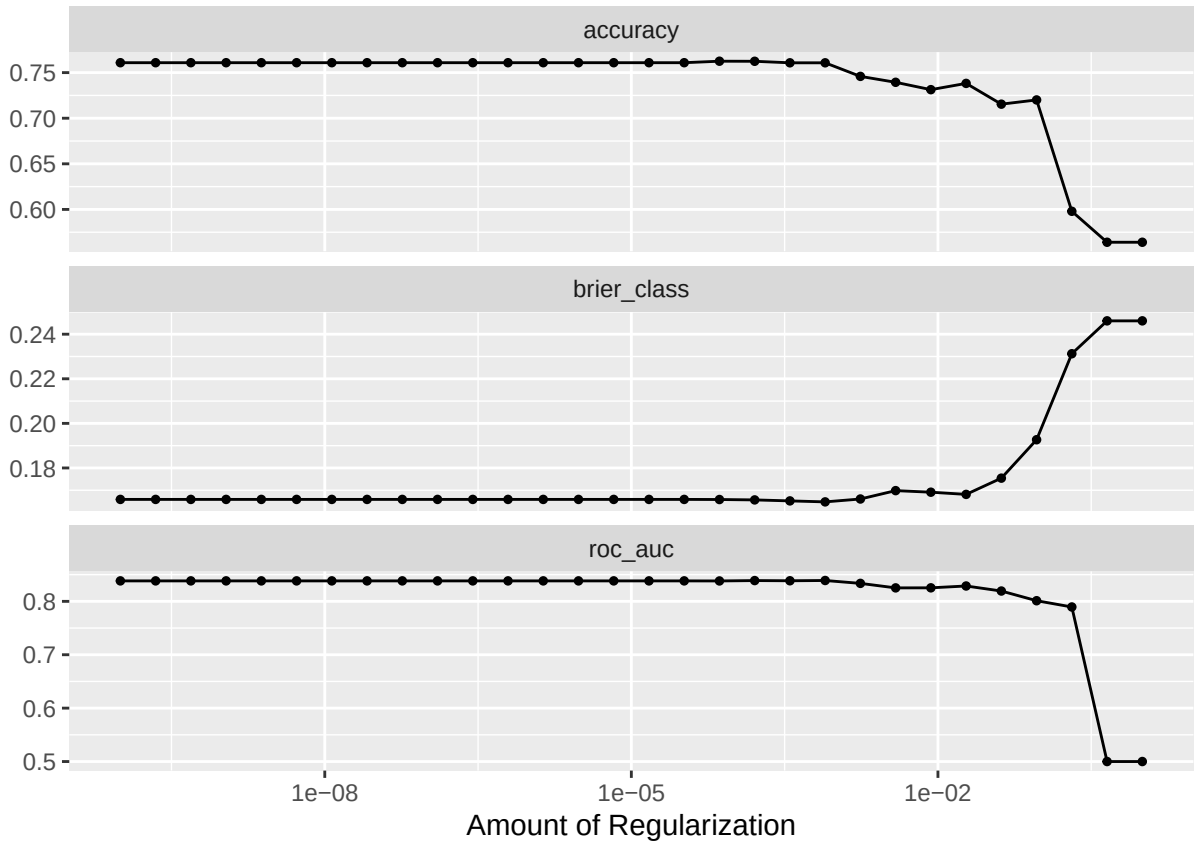
```
## 38 44 40.94 0.007638
## 39 45 41.37 0.006959
## 40 47 41.86 0.006341
## 41 48 42.32 0.005778
## 42 48 42.72 0.005264
## 43 49 43.08 0.004797
## 44 53 43.51 0.004371
## 45 53 43.91 0.003982
## 46 53 44.26 0.003629
##
## ...
## and 38 more lines.
```

Indicators, averages, and coefficients

Remove correlating

```
train_lasso(recipe_iac, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##    penalty .metric .estimator  mean     n std_err .config
##      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 7.88e- 4 roc_auc binary     0.839    10  0.0114 Preprocessor1_Model21
## 2 1.61e- 4 roc_auc binary     0.839    10  0.0114 Preprocessor1_Model19
## 3 3.56e- 4 roc_auc binary     0.839    10  0.0112 Preprocessor1_Model20
## 4 1   e-10 roc_auc binary     0.838    10  0.0116 Preprocessor1_Model01
## 5 2.21e-10 roc_auc binary     0.838    10  0.0116 Preprocessor1_Model02
## # A tibble: 5 x 7
##    penalty .metric  .estimator  mean     n std_err .config
##      <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 7.28e- 5 accuracy binary     0.763    10  0.0110 Preprocessor1_Model18
## 2 1.61e- 4 accuracy binary     0.762    10  0.0115 Preprocessor1_Model19
## 3 1   e-10 accuracy binary     0.761    10  0.0117 Preprocessor1_Model01
## 4 2.21e-10 accuracy binary     0.761    10  0.0117 Preprocessor1_Model02
## 5 4.89e-10 accuracy binary     0.761    10  0.0117 Preprocessor1_Model03
## Best accuracy:
## # A tibble: 1 x 2
##    penalty .config
##      <dbl> <chr>
## 1 0.000788 Preprocessor1_Model21
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.000788046281566992
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 39 x 3
##    term                                              estimate  penalty
##    <chr>                                                <dbl>    <dbl>
##  1 RuleCaseRepetition.max_repetition_frac              -36.3    0.000788
##  2 RuleTooFewVerbs.min_verb_frac                       -25.7    0.000788
##  3 RuleTooManyNominalConstructions.max_noun_frac       -11.1    0.000788
##  4 mattr                                                -7.85   0.000788
##  5 mattr.v                                              -6.93   0.000788
##  6 ttr                                                  -3.48   0.000788
##  7 RuleCaseRepetition.max_repetition_count.v            -3.09   0.000788
##  8 fkgl                                                 -1.41   0.000788
##  9 RuleTooManyNegations.max_allowable_negations.v       -1.21   0.000788
## 10 RuleTooManyNominalConstructions.max_allowable_nouns.v -1.21   0.000788
## 11 RuleInfVerbDistance.max_distance.v                  -0.789  0.000788
## 12 RuleTooManyNegations.max_negation_frac              -0.631  0.000788
## 13 RuleDoubleAdpos.max_allowable_distance.v            -0.397  0.000788
## 14 fre                                                 -0.301  0.000788
## 15 RulePredAtClauseBeginning.max_order.v               -0.226  0.000788
## 16 RulePredSubjDistance.max_distance.v                 -0.165  0.000788
## 17 RulePredSubjDistance.max_distance                   -0.0715 0.000788
## 18 hpoint                                              -0.0652 0.000788
## 19 RulePredObjDistance.max_distance                    -0.0638 0.000788
## 20 cli                                                 -0.0403 0.000788
## 21 RulePredObjDistance.max_distance.v                   0      0.000788
## 22 entropy                                              0      0.000788
## 23 RuleDoubleAdpos.max_allowable_distance               0.0180 0.000788
## 24 RulePredAtClauseBeginning.max_order                  0.0196 0.000788
## 25 RuleMultiPartVerbs.max_distance                      0.0200 0.000788
## 26 RuleCaseRepetition.max_repetition_count              0.0365 0.000788
## 27 verb_dist                                            0.0396 0.000788
## 28 RuleInfVerbDistance.max_distance                     0.0724 0.000788
## 29 RuleLongSentences.max_length                         0.131  0.000788
## 30 RuleTooManyNegations.max_allowable_negations         0.183  0.000788
## 31 RuleMultiPartVerbs.max_distance.v                    0.255  0.000788
## 32 RuleTooManyNominalConstructions.max_allowable_nouns  0.269  0.000788
## 33 RuleTooManyNegations.max_negation_frac.v             0.456  0.000788
## 34 RuleLongSentences.max_length.v                       1.72   0.000788
## 35 RuleTooManyNominalConstructions.max_noun_frac.v      2.39   0.000788
## 36 mamr                                                 4.27   0.000788
```

```
## 37 activity                                                 14.4    0.000788
## 38 maentropy.v                                              23.0    0.000788
## 39 (Intercept)                                              72.7    0.000788
```

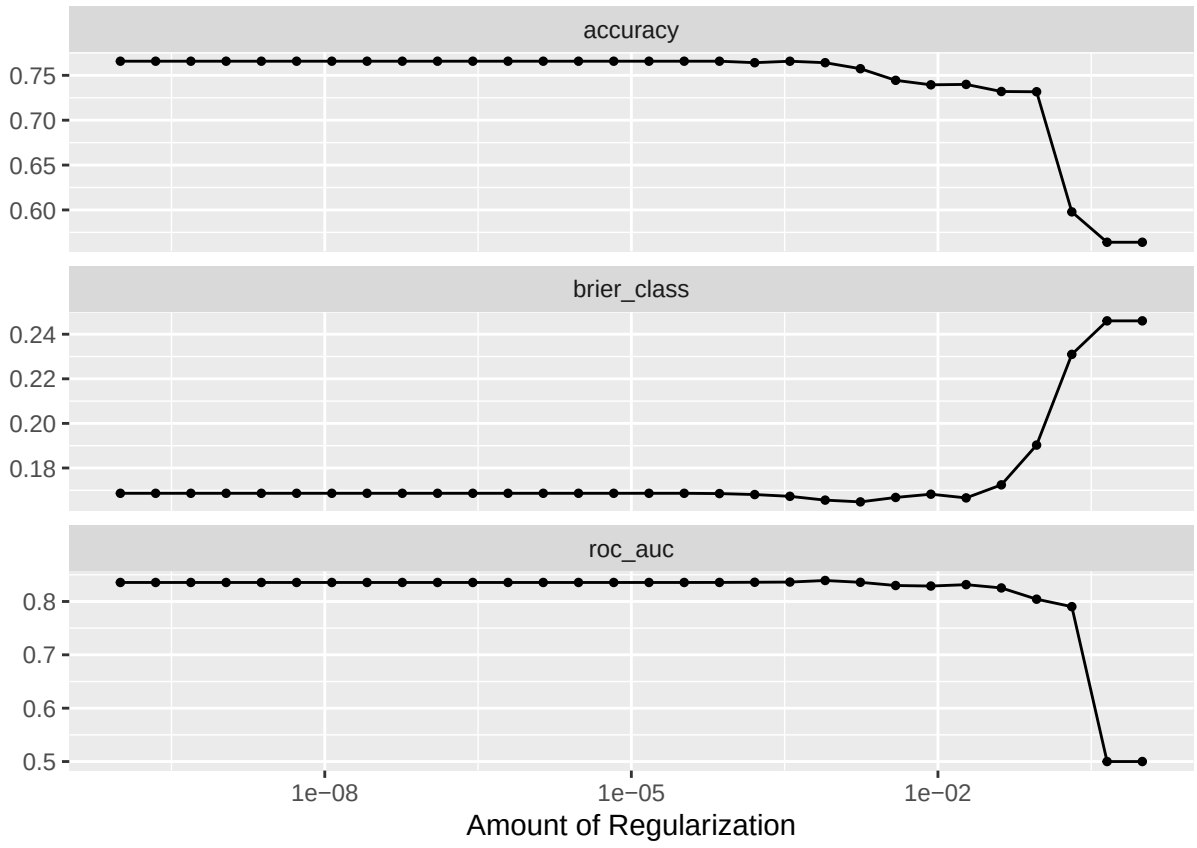
```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.238700
## 2   1  2.87 0.217500
## 3   1  5.27 0.198200
## 4   1  7.28 0.180600
## 5   1  8.99 0.164600
## 6   3 10.62 0.149900
## 7   4 12.26 0.136600
## 8   4 13.69 0.124500
## 9   4 14.91 0.113400
## 10  4 15.95 0.103300
## 11  3 16.85 0.094160
## 12  5 17.85 0.085800
## 13  6 18.86 0.078170
## 14  6 19.95 0.071230
## 15  7 21.03 0.064900
## 16  7 21.96 0.059140
## 17  8 22.83 0.053880
## 18  8 23.59 0.049100
## 19  8 24.24 0.044730
## 20 10 24.97 0.040760
## 21 11 25.71 0.037140
## 22 11 26.39 0.033840
## 23 11 26.97 0.030830
## 24 11 27.46 0.028090
## 25 14 27.93 0.025600
## 26 13 28.38 0.023320
## 27 13 28.77 0.021250
## 28 14 29.16 0.019360
## 29 15 29.51 0.017640
## 30 16 29.82 0.016080
## 31 17 30.11 0.014650
## 32 18 30.43 0.013350
## 33 18 30.71 0.012160
## 34 18 30.94 0.011080
## 35 20 31.23 0.010100
```

```
## 36 21 31.50 0.009200
## 37 21 31.77 0.008382
## 38 21 31.98 0.007638
## 39 23 32.18 0.006959
## 40 25 32.37 0.006341
## 41 27 32.62 0.005778
## 42 27 32.85 0.005264
## 43 29 33.06 0.004797
## 44 30 33.44 0.004371
## 45 31 33.88 0.003982
## 46 32 34.25 0.003629
##
## ...
## and 38 more lines.
```

Keep correlating

```
train_lasso(recipe_iac_nocorr, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##     penalty .metric .estimator  mean     n std_err .config
##       <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.000788  roc_auc binary     0.839    10 0.00954 Preprocessor1_Model21
## 2 0.000356  roc_auc binary     0.836    10 0.00997 Preprocessor1_Model20
## 3 0.000161  roc_auc binary     0.836    10 0.0101  Preprocessor1_Model19
## 4 0.00174   roc_auc binary     0.836    10 0.0120  Preprocessor1_Model22
## 5 0.0000728 roc_auc binary     0.836    10 0.00997 Preprocessor1_Model18
## # A tibble: 5 x 7
##    penalty .metric  .estimator  mean     n std_err .config
##      <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 1   e-10 accuracy binary     0.766    10  0.0102 Preprocessor1_Model01
## 2 2.21e-10 accuracy binary     0.766    10  0.0102 Preprocessor1_Model02
## 3 4.89e-10 accuracy binary     0.766    10  0.0102 Preprocessor1_Model03
## 4 1.08e- 9 accuracy binary     0.766    10  0.0102 Preprocessor1_Model04
## 5 2.40e- 9 accuracy binary     0.766    10  0.0102 Preprocessor1_Model05
## Best accuracy:
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <chr>
## 1 0.00174 Preprocessor1_Model22
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.00174332882219999
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 45 x 3
##    term                                                estimate penalty
##    <chr>                                                  <dbl>   <dbl>
##  1 RuleTooFewVerbs.min_verb_frac                        -24.0    0.00174
##  2 RuleTooManyNominalConstructions.max_noun_frac         -8.86   0.00174
##  3 mattr                                                 -7.30   0.00174
##  4 RuleCaseRepetition.max_repetition_count.v              -2.90   0.00174
##  5 ttr                                                   -1.73   0.00174
##  6 RuleTooManyNominalConstructions.max_allowable_nouns.v -1.26   0.00174
##  7 RuleTooManyNegations.max_allowable_negations.v        -0.946  0.00174
##  8 RuleInfVerbDistance.max_distance.v                    -0.804  0.00174
##  9 ari                                                   -0.457  0.00174
## 10 RuleDoubleAdpos.max_allowable_distance.v              -0.372  0.00174
## 11 smog                                                  -0.166  0.00174
## 12 RulePredAtClauseBeginning.max_order.v                 -0.157  0.00174
## 13 mattr.v                                               -0.146  0.00174
## 14 RulePredSubjDistance.max_distance.v                   -0.127  0.00174
## 15 entropy                                               -0.101  0.00174
## 16 fre                                                   -0.0976 0.00174
## 17 RulePredObjDistance.max_distance                      -0.0503 0.00174
## 18 gf                                                    -0.0500 0.00174
## 19 RulePredSubjDistance.max_distance                     -0.0450 0.00174
## 20 hpoint                                                -0.0390 0.00174
## 21 RuleTooManyNegations.max_negation_frac                 0      0.00174
## 22 RuleTooManyNegations.max_allowable_negations           0      0.00174
## 23 RuleCaseRepetition.max_repetition_count                0      0.00174
## 24 RuleCaseRepetition.max_repetition_frac                 0      0.00174
## 25 RulePredObjDistance.max_distance.v                     0      0.00174
## 26 cli                                                    0      0.00174
## 27 maentropy                                              0      0.00174
## 28 fkgl                                                   0      0.00174
## 29 RuleMultiPartVerbs.max_distance                        0.0115 0.00174
## 30 RulePredAtClauseBeginning.max_order                    0.0134 0.00174
## 31 RuleDoubleAdpos.max_allowable_distance                 0.0155 0.00174
## 32 verb_dist                                              0.0401 0.00174
## 33 RuleInfVerbDistance.max_distance                       0.0752 0.00174
## 34 RuleLongSentences.max_length                           0.0980 0.00174
## 35 RuleTooManyNominalConstructions.max_allowable_nouns    0.145  0.00174
## 36 RuleMultiPartVerbs.max_distance.v                      0.283  0.00174
## 37 RuleTooManyNegations.max_negation_frac.v               0.352  0.00174
```

```
## 38 mamr                                                  1.55   0.00174
## 39 RuleLongSentences.max_length.v                        1.61   0.00174
## 40 atl                                                   1.67   0.00174
## 41 RuleTooManyNominalConstructions.max_noun_frac.v       1.92   0.00174
## 42 (Intercept)                                           7.69   0.00174
## 43 maentropy.v                                          10.9    0.00174
## 44 activity                                             13.2    0.00174
## 45 RuleCaseRepetition.max_repetition_frac.v             26.2    0.00174
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.238700
## 2   1  2.87 0.217500
## 3   3  5.53 0.198200
## 4   3  7.92 0.180600
## 5   3  9.94 0.164600
## 6   3 11.67 0.149900
## 7   3 13.14 0.136600
## 8   4 14.47 0.124500
## 9   5 15.72 0.113400
## 10  5 16.94 0.103300
## 11  5 17.97 0.094160
## 12  6 19.24 0.085800
## 13  7 20.40 0.078170
## 14  7 21.47 0.071230
## 15  8 22.39 0.064900
## 16  6 23.22 0.059140
## 17  6 23.92 0.053880
## 18  7 24.53 0.049100
## 19  8 25.10 0.044730
## 20  9 25.78 0.040760
## 21 10 26.48 0.037140
## 22 10 27.09 0.033840
## 23 10 27.62 0.030830
## 24 11 28.13 0.028090
## 25 11 28.57 0.025600
## 26 12 28.95 0.023320
## 27 13 29.31 0.021250
## 28 15 29.68 0.019360
## 29 15 30.02 0.017640
## 30 16 30.34 0.016080
## 31 15 30.62 0.014650
```

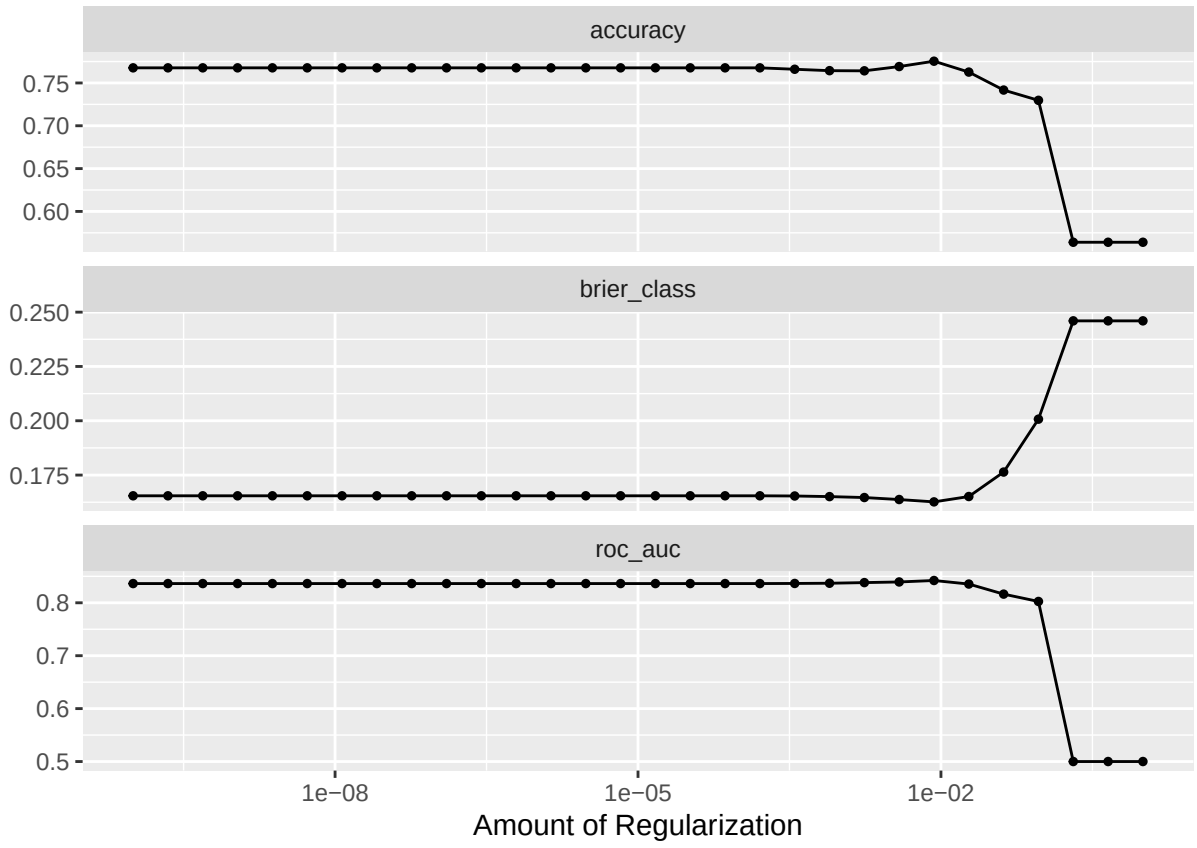
```
## 32 16 30.87 0.013350
## 33 17 31.12 0.012160
## 34 19 31.42 0.011080
## 35 22 31.72 0.010100
## 36 23 31.99 0.009200
## 37 25 32.32 0.008382
## 38 25 32.72 0.007638
## 39 26 33.07 0.006959
## 40 27 33.37 0.006341
## 41 29 33.65 0.005778
## 42 30 34.05 0.005264
## 43 30 34.48 0.004797
## 44 31 34.99 0.004371
## 45 31 35.42 0.003982
## 46 33 35.81 0.003629
##
## ...
## and 51 more lines.
```

Counts

Remove correlating

```
train_lasso(recipe_counts, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##    penalty .metric .estimator  mean     n std_err .config
##      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.00853  roc_auc binary     0.842    10  0.0105 Preprocessor1_Model24
## 2 0.00386  roc_auc binary     0.839    10  0.0112 Preprocessor1_Model23
## 3 0.00174  roc_auc binary     0.838    10  0.0117 Preprocessor1_Model22
## 4 0.000788 roc_auc binary     0.837    10  0.0119 Preprocessor1_Model21
## 5 0.000356 roc_auc binary     0.837    10  0.0123 Preprocessor1_Model20
## # A tibble: 5 x 7
##    penalty .metric  .estimator  mean     n std_err .config
##      <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 8.53e- 3 accuracy binary     0.775    10 0.00946 Preprocessor1_Model24
## 2 3.86e- 3 accuracy binary     0.769    10 0.0143  Preprocessor1_Model23
## 3 1   e-10 accuracy binary     0.768    10 0.0119  Preprocessor1_Model01
## 4 2.21e-10 accuracy binary     0.768    10 0.0119  Preprocessor1_Model02
## 5 4.89e-10 accuracy binary     0.768    10 0.0119  Preprocessor1_Model03
## Best accuracy:
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <chr>
## 1 0.00853 Preprocessor1_Model24
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.00853167852417281
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 33 x 3
##    term                            estimate penalty
##    <chr>                              <dbl>   <dbl>
##  1 RuleIncompleteConjunction     -677.       0.00853
##  2 RuleRelativisticExpressions   -479.       0.00853
##  3 RuleGPdeverbsubj              -182.       0.00853
##  4 RuleGPcoordovs                -146.       0.00853
##  5 RuleLiteraryStyle             -124.       0.00853
##  6 RulePassive                   -107.       0.00853
##  7 RuleGPdeverbaddr               -61.1      0.00853
##  8 RuleDoubleAdpos                -35.0      0.00853
##  9 RuleGPwordorder                 -5.62     0.00853
## 10 (Intercept)                     -1.76     0.00853
## 11 word_count                      -0.000863 0.00853
## 12 RuleGPpatinstr                   0        0.00853
## 13 RuleGPpatbenperson               0        0.00853
## 14 RuleReflexivePassWithAnimSubj    0        0.00853
## 15 RuleFunctionWordRepetition       0        0.00853
## 16 RuleWeakMeaningWords             0        0.00853
## 17 RuleAbstractNouns                0        0.00853
## 18 RuleConfirmationExpressions      0        0.00853
## 19 RuleRedundantExpressions         0        0.00853
## 20 RulePredObjDistance              0        0.00853
## 21 RuleDoubleComparison             0        0.00853
## 22 num_hapax                        0        0.00853
## 23 sent_count                       0.0128   0.00853
## 24 RuleWrongValencyCase             0.946    0.00853
## 25 RuleInfVerbDistance              1.15     0.00853
## 26 RuleVerbalNouns                  3.79     0.00853
## 27 RulePredSubjDistance            21.7      0.00853
## 28 RuleMultiPartVerbs              29.7      0.00853
## 29 RuleTooLongExpressions          83.4      0.00853
## 30 RuleGPadjective                184.       0.00853
## 31 RuleWrongVerbonominalCase      314.       0.00853
## 32 RuleAnaphoricReferences        353.       0.00853
## 33 RuleAmbiguousRegards           499.       0.00853
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
```

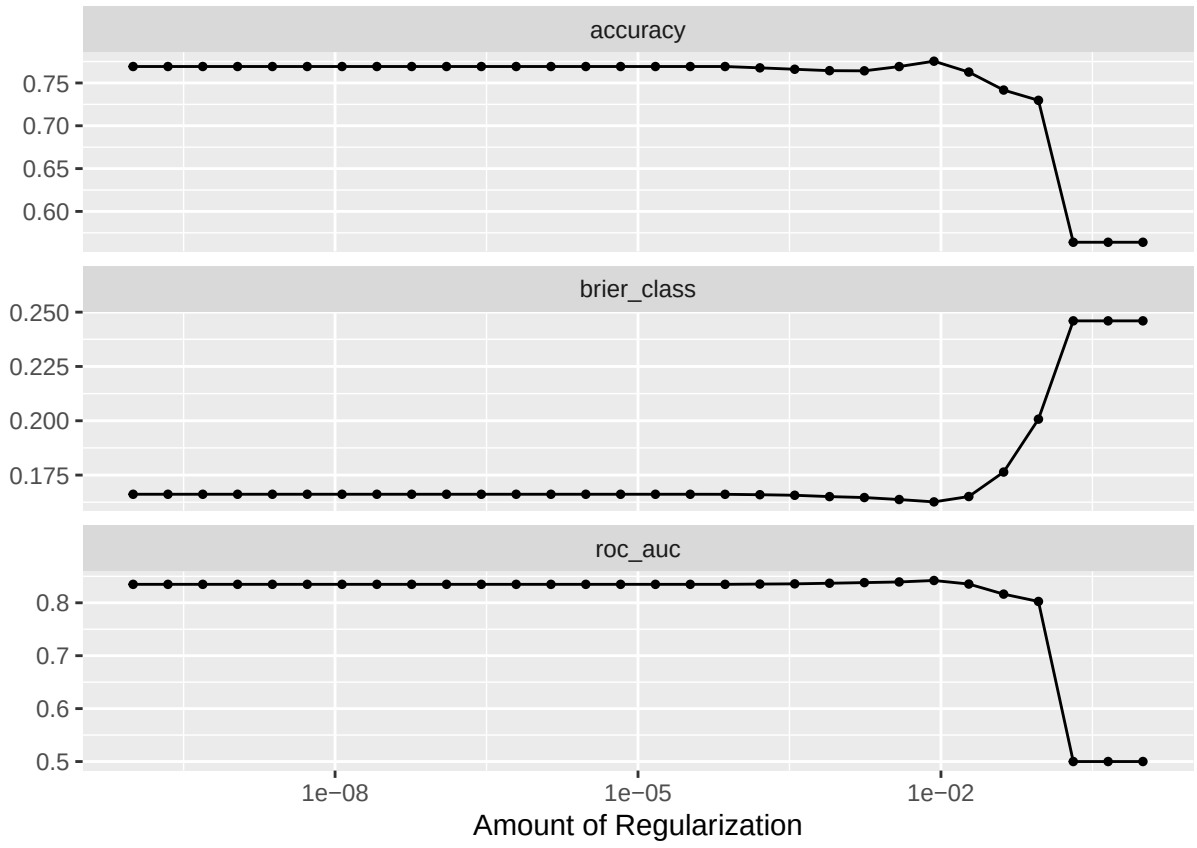
```
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model ----------------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.172400
## 2   2  2.42 0.157100
## 3   2  4.68 0.143100
## 4   4  7.41 0.130400
## 5   4  9.82 0.118800
## 6   4 11.88 0.108300
## 7   5 13.67 0.098640
## 8   5 15.21 0.089880
## 9   6 16.56 0.081900
## 10  6 17.97 0.074620
## 11  6 19.19 0.067990
## 12  6 20.24 0.061950
## 13  6 21.14 0.056450
## 14  6 21.92 0.051430
## 15  8 22.86 0.046860
## 16  9 23.79 0.042700
## 17 11 24.66 0.038910
## 18 13 25.60 0.035450
## 19 13 26.44 0.032300
## 20 14 27.22 0.029430
## 21 14 27.92 0.026820
## 22 16 28.59 0.024440
## 23 16 29.42 0.022260
## 24 16 30.15 0.020290
## 25 17 30.79 0.018480
## 26 18 31.36 0.016840
## 27 18 31.88 0.015350
## 28 18 32.33 0.013980
## 29 18 32.71 0.012740
## 30 19 33.05 0.011610
## 31 19 33.37 0.010580
## 32 19 33.64 0.009638
## 33 20 33.88 0.008782
## 34 21 34.10 0.008001
## 35 23 34.30 0.007291
## 36 24 34.47 0.006643
## 37 25 34.64 0.006053
## 38 26 34.78 0.005515
## 39 26 34.90 0.005025
## 40 27 35.03 0.004579
## 41 27 35.13 0.004172
```

```
## 42 28 35.22 0.003801
## 43 28 35.30 0.003464
## 44 29 35.36 0.003156
## 45 30 35.43 0.002876
## 46 30 35.48 0.002620
##
## ...
## and 23 more lines.
```

Keep correlating

```
train_lasso(recipe_counts_nocorr, training_set, folds)
```

```
## Lasso tune workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = tune()
##   mixture = 1
##
## Computational engine: glmnet
##
## Lasso tuning metrics:
```




```
## # A tibble: 5 x 7
##    penalty .metric .estimator  mean     n std_err .config
##      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 0.00853  roc_auc binary     0.842    10  0.0105 Preprocessor1_Model24
## 2 0.00386  roc_auc binary     0.839    10  0.0112 Preprocessor1_Model23
## 3 0.00174  roc_auc binary     0.838    10  0.0117 Preprocessor1_Model22
## 4 0.000788 roc_auc binary     0.837    10  0.0119 Preprocessor1_Model21
## 5 0.000356 roc_auc binary     0.836    10  0.0118 Preprocessor1_Model20
## # A tibble: 5 x 7
##    penalty .metric  .estimator  mean     n std_err .config
##      <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 8.53e- 3 accuracy binary     0.775    10 0.00946 Preprocessor1_Model24
## 2 1   e-10 accuracy binary     0.769    10 0.0111  Preprocessor1_Model01
## 3 2.21e-10 accuracy binary     0.769    10 0.0111  Preprocessor1_Model02
## 4 4.89e-10 accuracy binary     0.769    10 0.0111  Preprocessor1_Model03
## 5 1.08e- 9 accuracy binary     0.769    10 0.0111  Preprocessor1_Model04
## Best accuracy:
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <chr>
## 1 0.00853 Preprocessor1_Model24
## Final workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
```

```
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model ---------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 0.00853167852417281
##   mixture = 1
##
## Computational engine: glmnet
##
## Final coefficients:
## # A tibble: 35 x 3
##    term                            estimate penalty
##    <chr>                              <dbl>   <dbl>
##  1 RuleIncompleteConjunction     -677.       0.00853
##  2 RuleRelativisticExpressions   -479.       0.00853
##  3 RuleGPdeverbsubj              -182.       0.00853
##  4 RuleGPcoordovs                -146.       0.00853
##  5 RuleLiteraryStyle             -124.       0.00853
##  6 RulePassive                   -107.       0.00853
##  7 RuleGPdeverbaddr               -61.1      0.00853
##  8 RuleDoubleAdpos                -35.0      0.00853
##  9 RuleGPwordorder                 -5.62     0.00853
## 10 (Intercept)                     -1.76     0.00853
## 11 word_count                      -0.000863 0.00853
## 12 RuleGPpatinstr                   0        0.00853
## 13 RuleGPpatbenperson               0        0.00853
## 14 RuleReflexivePassWithAnimSubj    0        0.00853
## 15 RuleFunctionWordRepetition       0        0.00853
## 16 RuleWeakMeaningWords             0        0.00853
## 17 RuleAbstractNouns                0        0.00853
## 18 RuleConfirmationExpressions      0        0.00853
## 19 RuleRedundantExpressions         0        0.00853
## 20 RulePredObjDistance              0        0.00853
## 21 RuleDoubleComparison             0        0.00853
## 22 syllab_count                     0        0.00853
## 23 char_count                       0        0.00853
## 24 num_hapax                        0        0.00853
## 25 sent_count                       0.0128   0.00853
## 26 RuleWrongValencyCase             0.946    0.00853
## 27 RuleInfVerbDistance              1.15     0.00853
## 28 RuleVerbalNouns                  3.79     0.00853
## 29 RulePredSubjDistance            21.7      0.00853
## 30 RuleMultiPartVerbs              29.7      0.00853
## 31 RuleTooLongExpressions          83.4      0.00853
## 32 RuleGPadjective                184.       0.00853
## 33 RuleWrongVerbonominalCase      314.       0.00853
## 34 RuleAnaphoricReferences        353.       0.00853
## 35 RuleAmbiguousRegards           499.       0.00853

## == Workflow [trained] ==========================================================
```

```
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------
##
## Call:  glmnet::glmnet(x = maybe_matrix(x), y = y, family = "binomial",      alpha = ~1)
##
##    Df  %Dev   Lambda
## 1   0  0.00 0.172400
## 2   2  2.42 0.157100
## 3   2  4.68 0.143100
## 4   4  7.41 0.130400
## 5   4  9.82 0.118800
## 6   4 11.88 0.108300
## 7   5 13.67 0.098640
## 8   5 15.21 0.089880
## 9   6 16.56 0.081900
## 10  6 17.97 0.074620
## 11  6 19.19 0.067990
## 12  6 20.24 0.061950
## 13  6 21.14 0.056450
## 14  6 21.92 0.051430
## 15  8 22.86 0.046860
## 16  9 23.79 0.042700
## 17 11 24.66 0.038910
## 18 13 25.60 0.035450
## 19 13 26.44 0.032300
## 20 14 27.22 0.029430
## 21 14 27.92 0.026820
## 22 16 28.59 0.024440
## 23 16 29.42 0.022260
## 24 16 30.15 0.020290
## 25 17 30.79 0.018480
## 26 18 31.36 0.016840
## 27 18 31.88 0.015350
## 28 18 32.33 0.013980
## 29 18 32.71 0.012740
## 30 19 33.05 0.011610
## 31 19 33.37 0.010580
## 32 19 33.64 0.009638
## 33 20 33.88 0.008782
## 34 21 34.10 0.008001
## 35 23 34.30 0.007291
## 36 24 34.47 0.006643
## 37 25 34.64 0.006053
## 38 26 34.78 0.005515
## 39 26 34.90 0.005025
## 40 27 35.03 0.004579
## 41 27 35.13 0.004172
```

```
## 42 28 35.22 0.003801
## 43 28 35.30 0.003464
## 44 29 35.36 0.003156
## 45 30 35.43 0.002876
## 46 30 35.48 0.002620
##
## ...
## and 38 more lines.
```

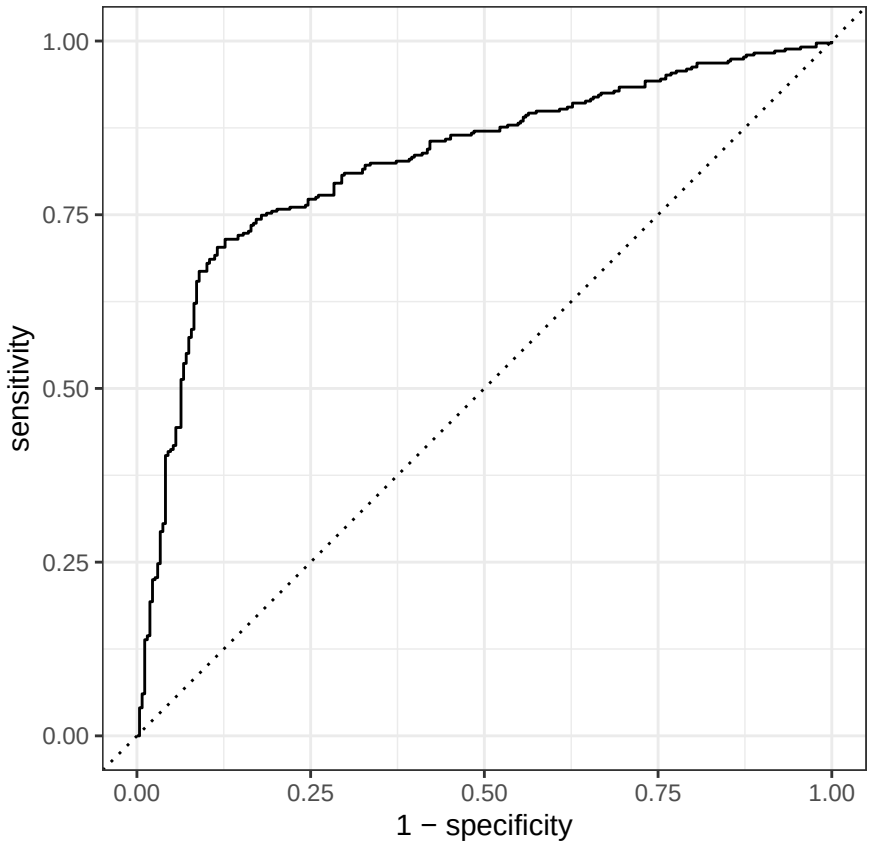
SVM

All variables

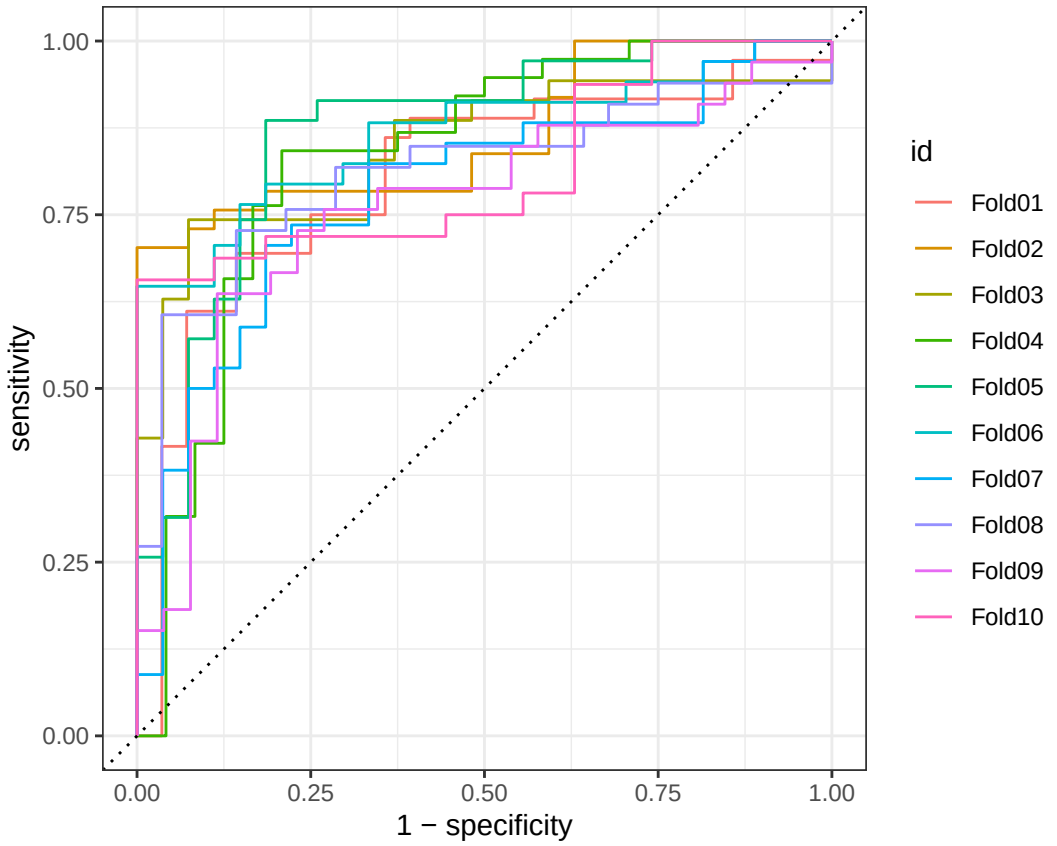
Remove correlating

```
train_svm(recipe_all, training_set, folds)
```

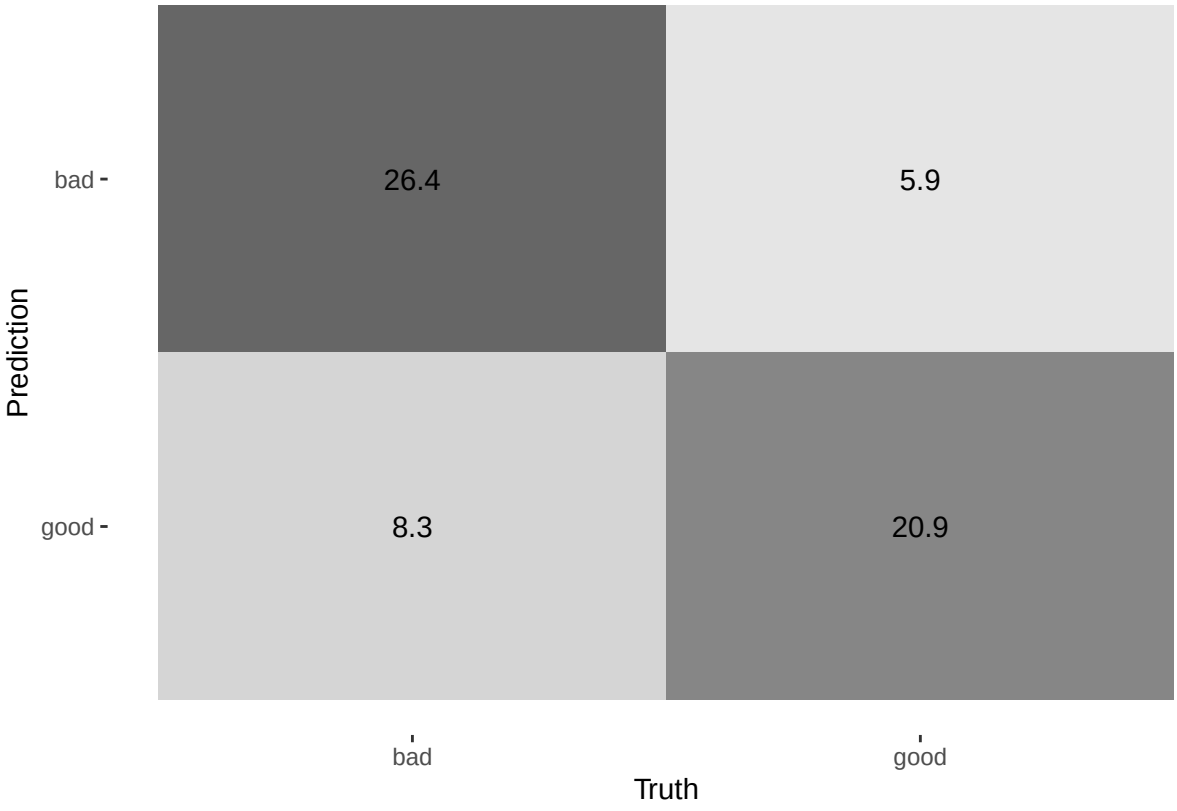
```
## SVM workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: svm_linear()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Linear Support Vector Machine Model Specification (classification)
##
## Computational engine: kernlab
##
## SVM metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n std_err .config
##   <chr>       <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy    binary     0.769    10 0.0121  Preprocessor1_Model1
## 2 brier_class binary     0.174    10 0.00527 Preprocessor1_Model1
## 3 roc_auc     binary     0.825    10 0.0118  Preprocessor1_Model1
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity

```
## [1] "\n"
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity
id
Fold01
Fold02
Fold03
Fold04
Fold05
Fold06
Fold07
Fold08
Fold09
Fold10

```
## [1] "\n"
```

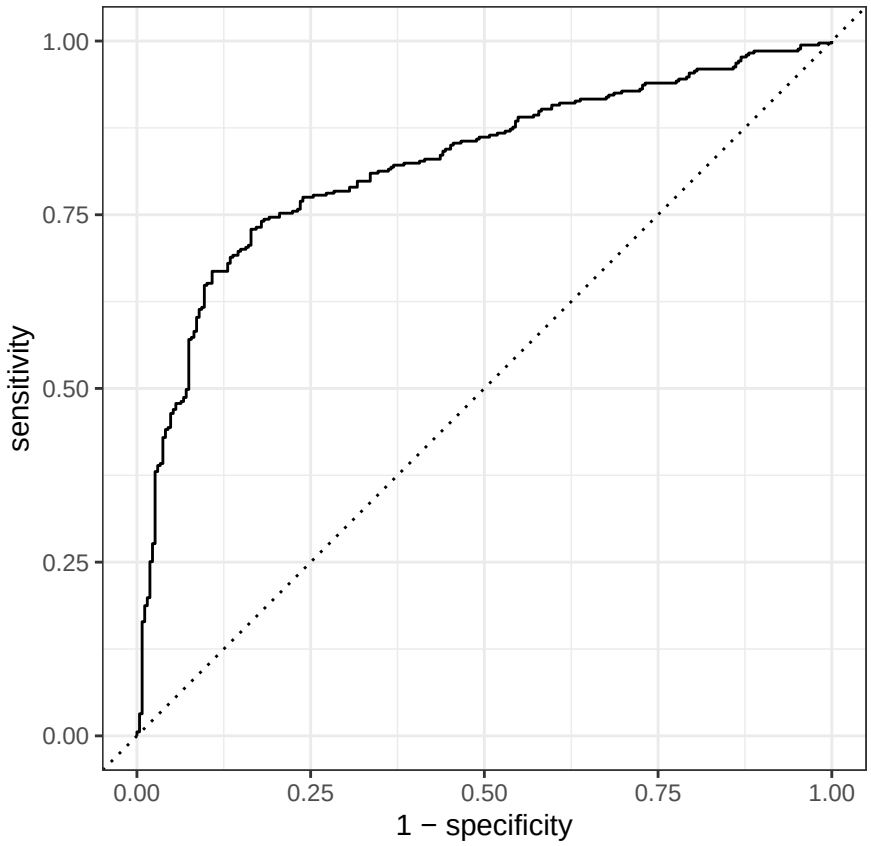

bad
good
Prediction
26.4
5.9
8.3
20.9
bad
good
Truth

```
## [1] "\n"
##  Setting default kernel parameters
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: svm_linear()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 286
##
## Objective Function Value : -242.3345
## Training error : 0.164228
## Probability model included.
```

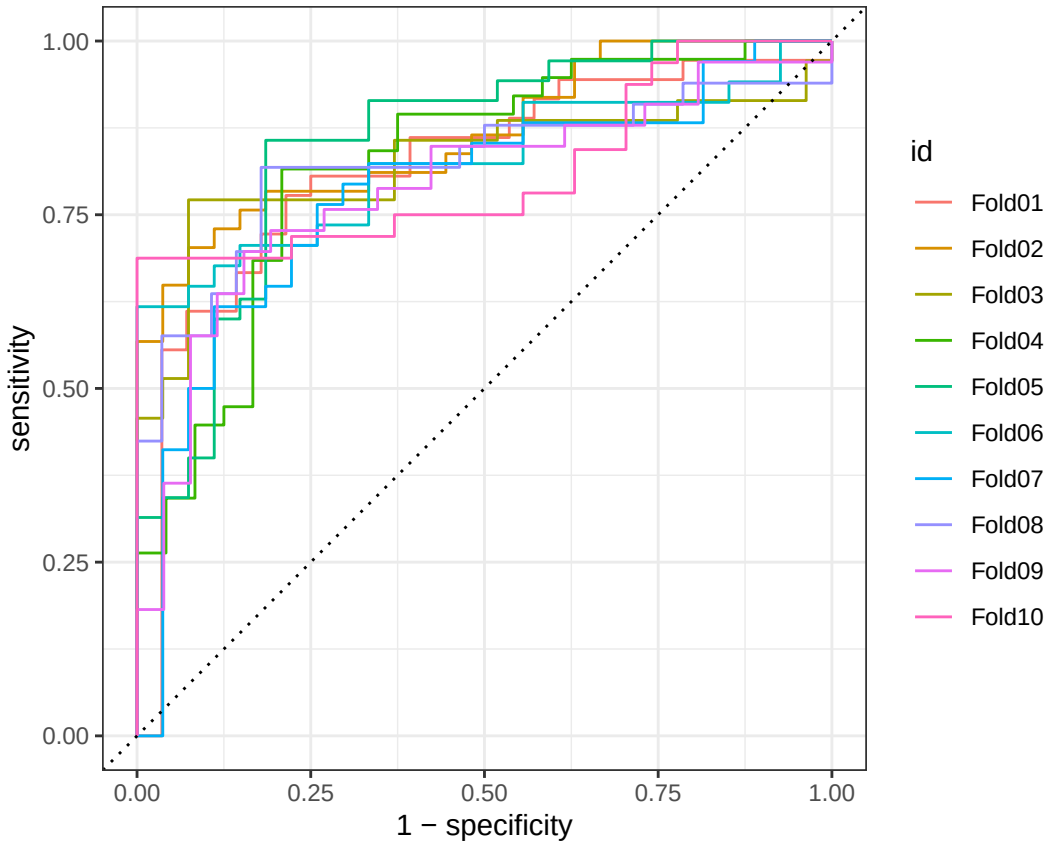
Keep correlating

```
train_svm(recipe_all_nocorr, training_set, folds)
```

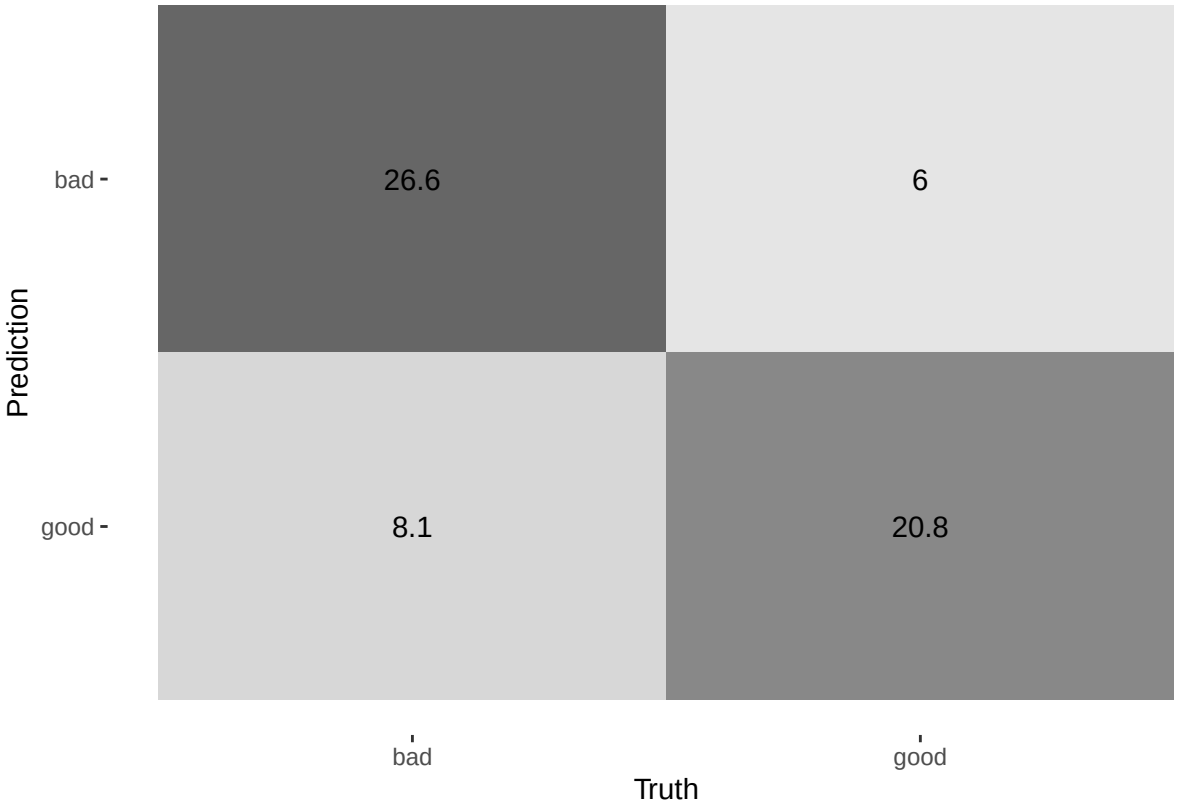
```
## SVM workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: svm_linear()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Linear Support Vector Machine Model Specification (classification)
##
## Computational engine: kernlab
##
## SVM metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n std_err .config
##   <chr>       <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy    binary     0.770    10 0.0136  Preprocessor1_Model1
## 2 brier_class binary     0.174    10 0.00481 Preprocessor1_Model1
## 3 roc_auc     binary     0.822    10 0.00777 Preprocessor1_Model1
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity

```
## [1] "\n"
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity
id
Fold01
Fold02
Fold03
Fold04
Fold05
Fold06
Fold07
Fold08
Fold09
Fold10

```
## [1] "\n"
```


bad
good
Prediction
26.6
6
8.1
20.8
bad
good
Truth

```
## [1] "\n"
##  Setting default kernel parameters
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: svm_linear()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 279
##
## Objective Function Value : -236.7433
## Training error : 0.164228
## Probability model included.
```

Random forest

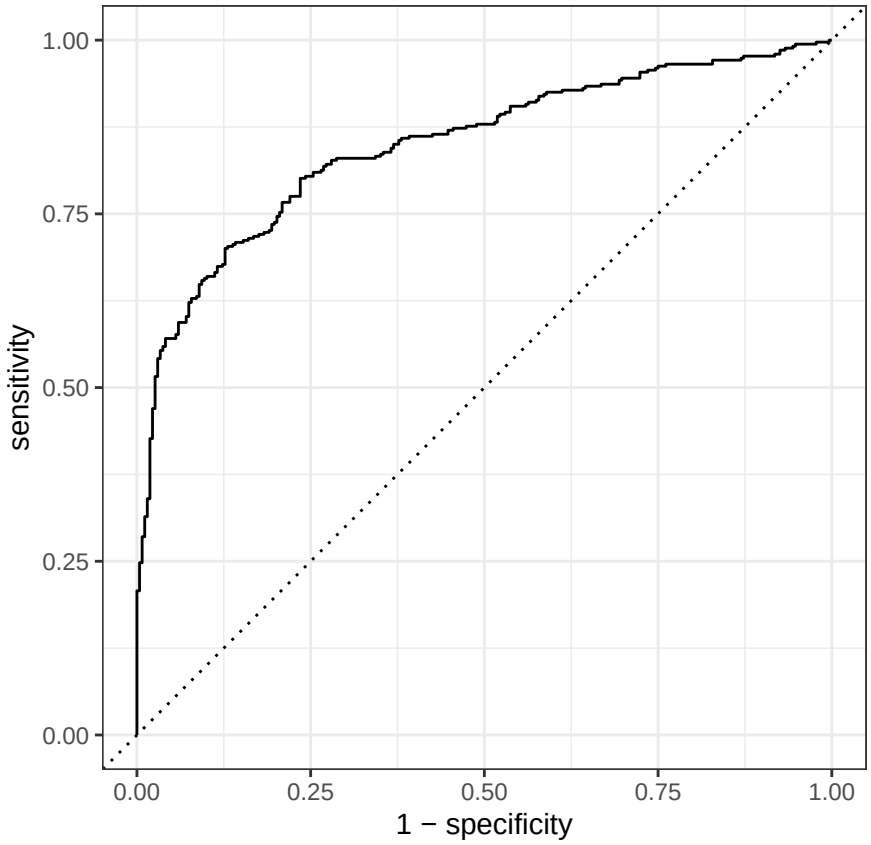
All variables

Remove correlating

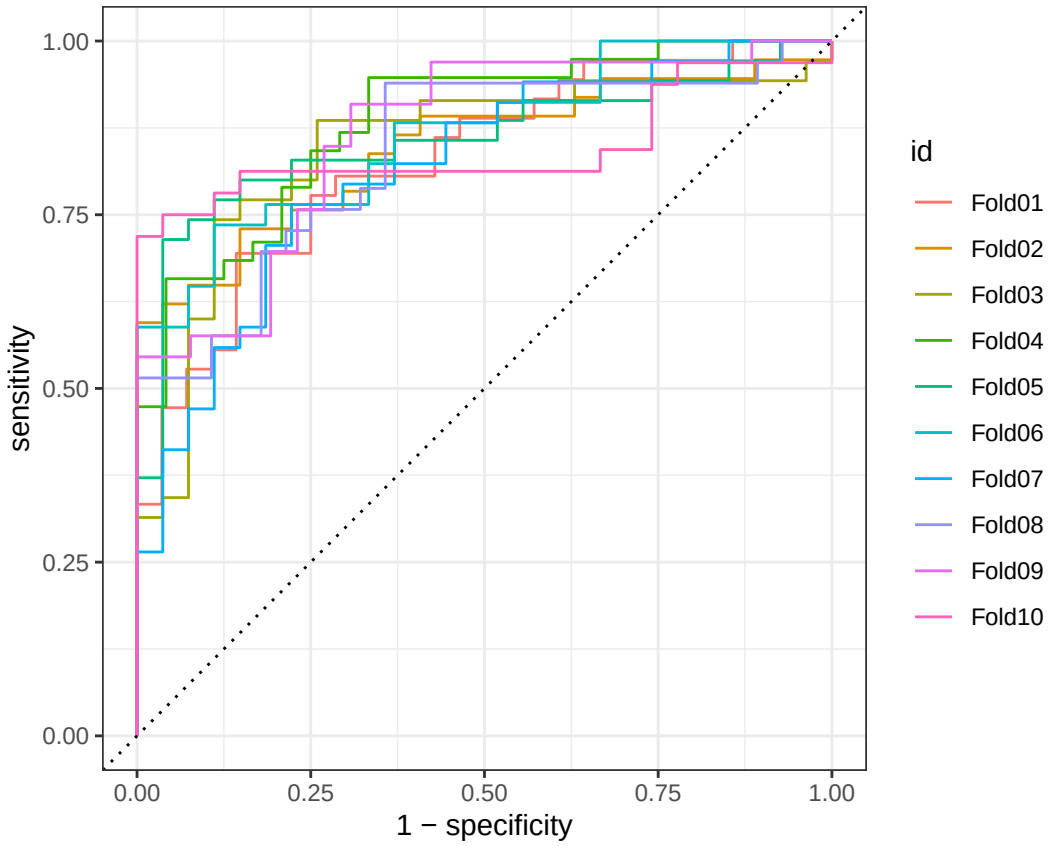
```
train_random_forest(recipe_all, training_set, folds)
```

```
## RF workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_normalize()
## * step_corr()
##
## -- Model -----------------------------------------------------------------------
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   trees = 1000
##
## Engine-Specific Arguments:
##   importance = impurity
##
```

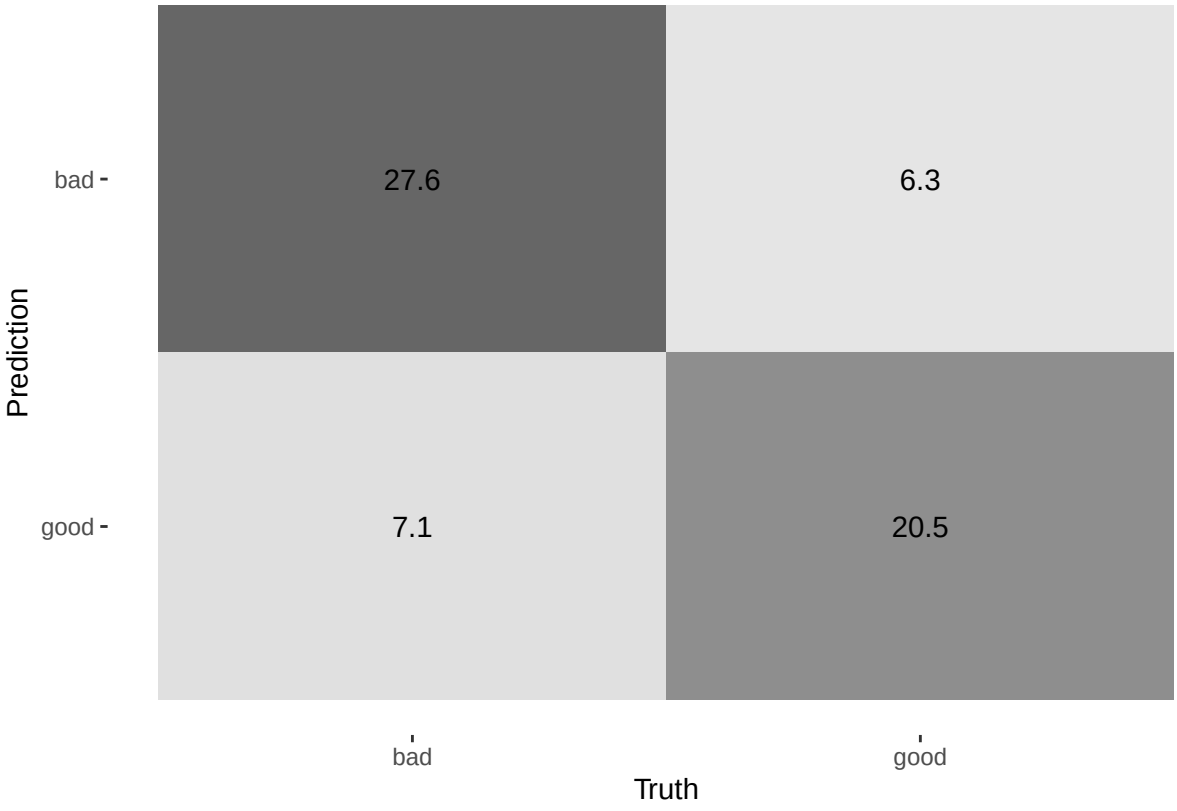
```
## Computational engine: ranger
##
## RF metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n std_err .config
##   <chr>       <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy    binary     0.782    10 0.00912 Preprocessor1_Model1
## 2 brier_class binary     0.156    10 0.00327 Preprocessor1_Model1
## 3 roc_auc     binary     0.849    10 0.00621 Preprocessor1_Model1
```




```
## [1] "\n"
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity
id
Fold01
Fold02
Fold03
Fold04
Fold05
Fold06
Fold07
Fold08
Fold09
Fold10

```
## [1] "\n"
```


bad
good
Prediction
27.6
6.3
7.1
20.5
bad
good
Truth

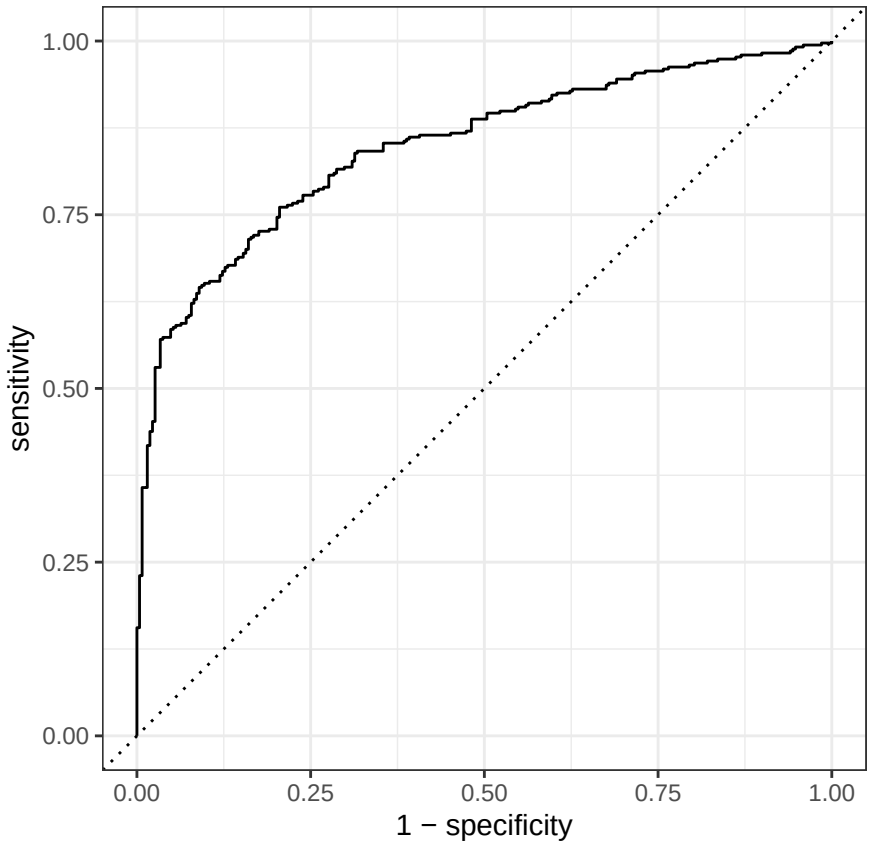
```
## [1] "\n"
```

```
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 5
##    splits           id     .metrics         .notes           .predictions
##    <list>           <chr>  <list>           <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [64 x 6]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [64 x 6]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [59 x 6]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [59 x 6]>
```

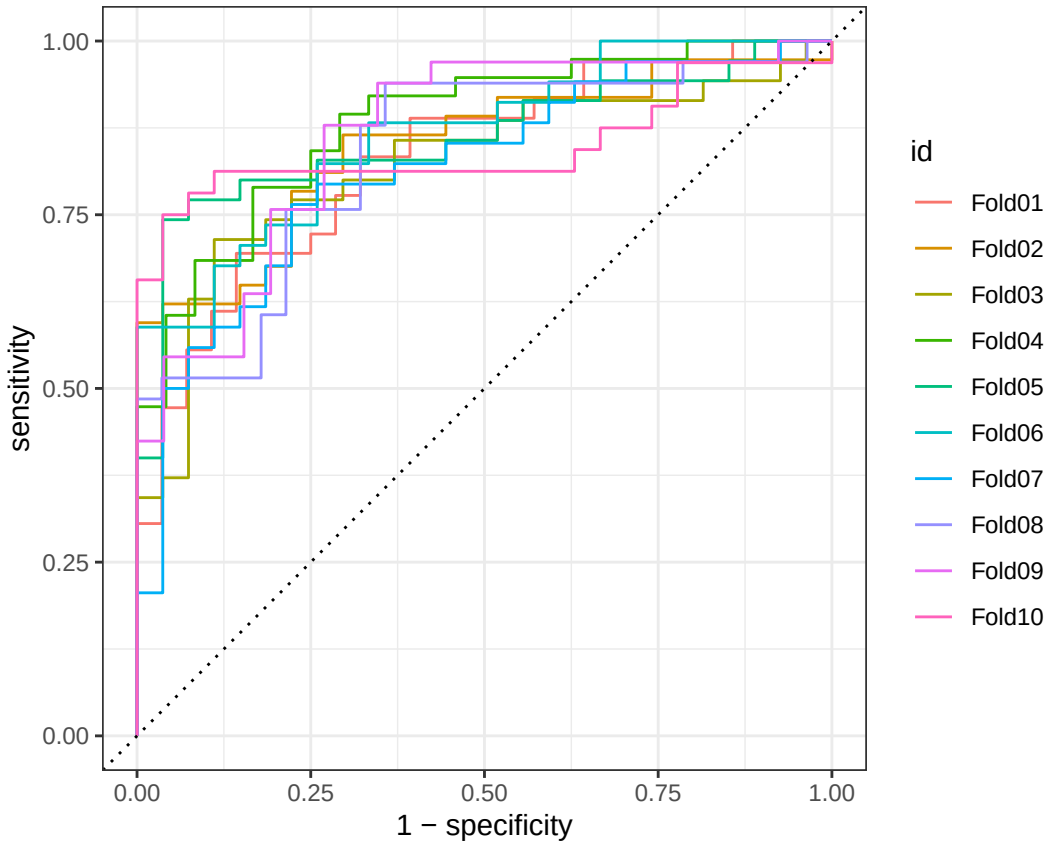
Keep correlating

```
train_random_forest(recipe_all_nocorr, training_set, folds)
```

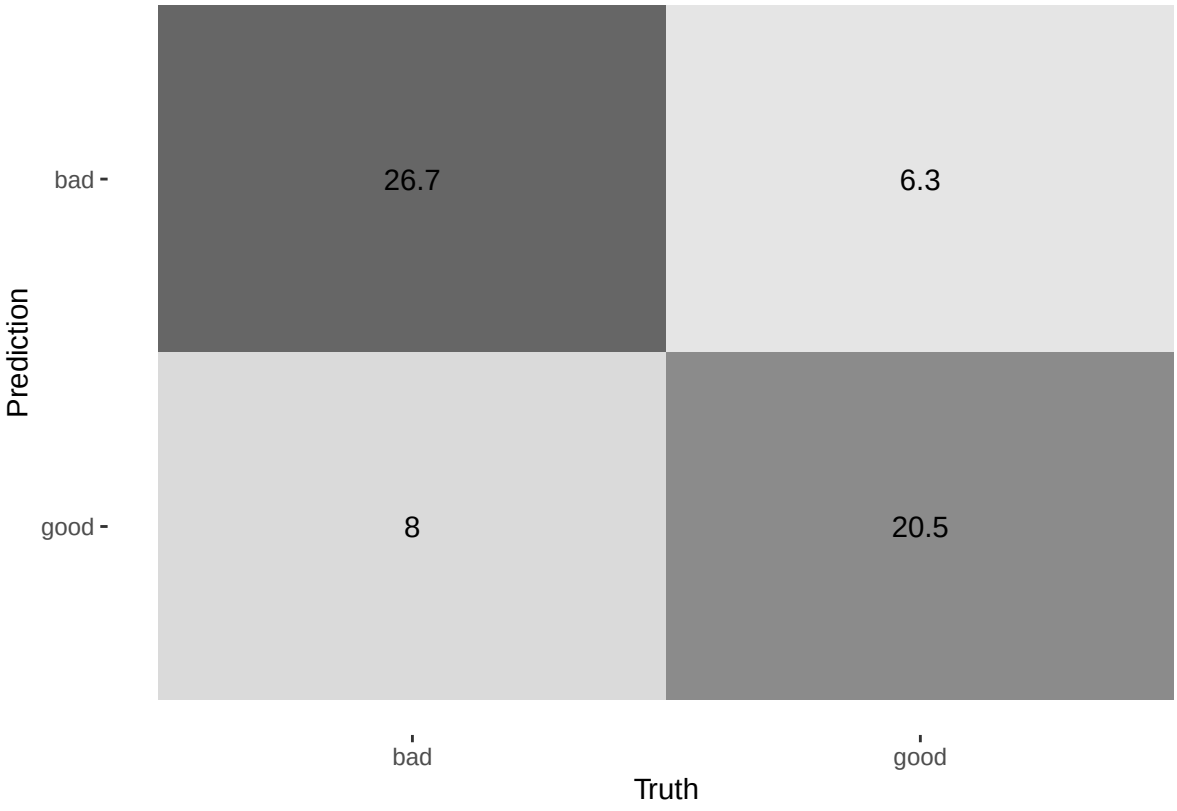
```
## RF workflow:
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   trees = 1000
##
## Engine-Specific Arguments:
##   importance = impurity
##
## Computational engine: ranger
##
## RF metrics:
## # A tibble: 3 x 6
##   .metric     .estimator  mean     n std_err .config
##   <chr>       <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy    binary     0.768    10 0.0119  Preprocessor1_Model1
## 2 brier_class binary     0.157    10 0.00381 Preprocessor1_Model1
## 3 roc_auc     binary     0.848    10 0.00685 Preprocessor1_Model1
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity

```
## [1] "\n"
```


1.00
0.75
0.50
0.25
0.00
sensitivity
0.00
0.25
0.50
0.75
1.00
1 − specificity
id
Fold01
Fold02
Fold03
Fold04
Fold05
Fold06
Fold07
Fold08
Fold09
Fold10

```
## [1] "\n"
```


bad
good
Prediction
26.7
6.3
8
20.5
bad
good
Truth

```
## [1] "\n"
```

```
## # Resampling results
## # 10-fold cross-validation using stratification
## # A tibble: 10 x 5
##    splits           id     .metrics         .notes           .predictions
##    <list>           <chr>  <list>           <list>           <list>
##  1 <split [551/64]> Fold01 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [64 x 6]>
##  2 <split [551/64]> Fold02 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [64 x 6]>
##  3 <split [553/62]> Fold03 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  4 <split [553/62]> Fold04 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  5 <split [553/62]> Fold05 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [62 x 6]>
##  6 <split [554/61]> Fold06 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  7 <split [554/61]> Fold07 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  8 <split [554/61]> Fold08 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [61 x 6]>
##  9 <split [556/59]> Fold09 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [59 x 6]>
## 10 <split [556/59]> Fold10 <tibble [3 x 4]> <tibble [0 x 3]> <tibble [59 x 6]>
```