Importance measures

```
set.seed(42)

library(rcompanion) # KW effect size calculation
library(rstatix) # Wilcox effect size calculation
```

```
##
## Attaching package: 'rstatix'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

```
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(QuantPsyc) # for the multivariate normality test
```

```
## Loading required package: boot
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:igraph':
##
##     as_data_frame, groups, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Loading required package: purrr
```

```
##
## Attaching package: 'purrr'
```

```
## The following objects are masked from 'package:igraph':
##
##     compose, simplify
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## The following object is masked from 'package:rstatix':
##
##     select
```

```
##
## Attaching package: 'QuantPsyc'
```

```
## The following object is masked from 'package:base':
##
##     norm
```

```
library(dunn.test)
library(nFactors) # for the scree plot
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
##
## Attaching package: 'nFactors'
```

```
## The following object is masked from 'package:lattice':
##
##     parallel
```

```
library(psych) # for PA FA
```

```
##
## Attaching package: 'psych'
```

```
## The following object is masked from 'package:boot':
##
##     logit
```

```
## The following object is masked from 'package:rcompanion':
##
##     phi
```

```
library(caret) # highly correlated features removal
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
##
## Attaching package: 'caret'
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 1.2.0 --
## v broom        1.0.5     v tibble       3.2.1
## v dials        1.3.0     v tidyr        1.3.1
## v infer        1.0.7     v tune         1.2.1
## v modeldata    1.4.0     v workflows    1.1.4
## v parsnip      1.2.1     v workflowsets 1.1.0
## v recipes      1.1.0     v yardstick    1.3.2
## v rsample      1.2.1
## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x ggplot2::%+%()           masks psych::%+%()
## x yardstick::accuracy()    masks rcompanion::accuracy()
## x scales::alpha()          masks ggplot2::alpha(), psych::alpha()
## x tibble::as_data_frame()  masks dplyr::as_data_frame(), igraph::as_data_frame()
## x infer::chisq_test()      masks rstatix::chisq_test()
## x purrr::compose()         masks igraph::compose()
## x tidyr::crossing()        masks igraph::crossing()
## x dials::degree()          masks igraph::degree()
## x scales::discard()        masks purrr::discard()
## x dplyr::filter()          masks rstatix::filter(), stats::filter()
## x dials::get_n()           masks rstatix::get_n()
## x dplyr::lag()             masks stats::lag()
## x caret::lift()            masks purrr::lift()
## x dials::neighbors()       masks igraph::neighbors()
## x yardstick::precision()   masks caret::precision()
## x infer::prop_test()       masks rstatix::prop_test()
## x yardstick::recall()      masks caret::recall()
## x MASS::select()           masks dplyr::select(), rstatix::select()
## x yardstick::sensitivity() masks caret::sensitivity()
## x purrr::simplify()        masks igraph::simplify()
## x yardstick::specificity() masks caret::specificity()
## x recipes::step()          masks stats::step()
## x infer::t_test()          masks rstatix::t_test()
## * Search for functions across packages at https://www.tidymodels.org/find/
```

```
library(vip)
```

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

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v lubridate 1.9.3     v stringr   1.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::%--%()       masks igraph::%--%()
## x ggplot2::%+%()          masks psych::%+%()
## x scales::alpha()         masks ggplot2::alpha(), psych::alpha()
## x tibble::as_data_frame() masks dplyr::as_data_frame(), igraph::as_data_frame()
## x readr::col_factor()     masks scales::col_factor()
## x purrr::compose()        masks igraph::compose()
## x tidyr::crossing()       masks igraph::crossing()
## x scales::discard()       masks purrr::discard()
## x dplyr::filter()         masks rstatix::filter(), stats::filter()
## x stringr::fixed()        masks recipes::fixed()
## x dplyr::lag()            masks stats::lag()
## x caret::lift()           masks purrr::lift()
## x MASS::select()          masks dplyr::select(), rstatix::select()
## x purrr::simplify()       masks igraph::simplify()
## x readr::spec()           masks yardstick::spec()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(paletteer) # color palettes

library(conflicted) # to resolve QuantPsyc x dplyr conflicts
conflict_prefer("select", "dplyr")
```

```
## [conflicted] Will prefer dplyr::select over any other package.
```

```
conflict_prefer("filter", "dplyr")
```

```
## [conflicted] Will prefer dplyr::filter over any other package.
```

Load and tidy data

```
pretty_names <- read_csv("../feat_name_mapping.csv")
```

```
## Rows: 85 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): name_orig, name_pretty
##
## 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.
```

```
prettify_feat_name <- function(x) {
  name <- pull(pretty_names %>%
    filter(name_orig == x), name_pretty)
  if (length(name) == 1) {
    return(name)
  } else {
    return(x)
  }
```

```
}

prettify_feat_name_vector <- function(x) {
  map(
    x,
    prettify_feat_name
  ) %>% unlist()
}


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

```
## Rows: 753 Columns: 108
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (20): fpath, KUK_ID, FileName, FileFormat, FolderPath, subcorpus, Source...
## dbl (85): RuleAbstractNouns, RuleAmbiguousRegards, RuleAnaphoricReferences, ...
## lgl  (3): ClarityPursuit, SyllogismBased, Bindingness
##
## 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.
```

```
.firstnonmetacolumn <- 17

data_no_nas <- 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.v = 0,
  RuleAmbiguousRegards = 0,
  RuleReflexivePassWithAnimSubj = 0,
  RuleTooManyNegations = 0,
  RuleTooManyNegations.max_negation_frac.v = 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.v = 0,
  RulePredObjDistance = 0,
  RulePredObjDistance.max_distance.v = 0,
  RuleInfVerbDistance = 0,
  RuleInfVerbDistance.max_distance.v = 0,
  RuleMultiPartVerbs = 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
)) %>%
# replace NAs with medians
mutate(across(c(
  RuleDoubleAdpos.max_allowable_distance,
  RuleTooManyNegations.max_negation_frac,
  RuleTooManyNegations.max_allowable_negations,
  RulePredSubjDistance.max_distance,
```

```
    RulePredObjDistance.max_distance,
    RuleInfVerbDistance.max_distance,
    RuleMultiPartVerbs.max_distance
  ), ~ coalesce(., median(., na.rm = TRUE)))) %>%
  # merge GPs
  mutate(
    GPs = RuleGPcoordovs +
      RuleGPdeverbaddr +
      RuleGPpatinstr +
      RuleGPdeverbsubj +
      RuleGPadjective +
      RuleGPpatbenperson +
      RuleGPwordorder
  ) %>%
  select(!c(
    RuleGPcoordovs,
    RuleGPdeverbaddr,
    RuleGPpatinstr,
    RuleGPdeverbsubj,
    RuleGPadjective,
    RuleGPpatbenperson,
    RuleGPwordorder
  ))

data_clean <- data_no_nas %>%
  # norm data expected to correlate with text length
  mutate(across(c(
    GPs,
    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 text-length dependent
  select(!c(
    RuleTooFewVerbs,
    RuleTooManyNegations,
    RuleTooManyNominalConstructions,
    RuleCaseRepetition,
    RuleLongSentences,
    RulePredAtClauseBeginning,
    syllab_count,
    char_count
  )) %>%
  # remove variables identified as unreliable
  select(!c(
    RuleAmbiguousRegards,
    RuleFunctionWordRepetition,
    RuleDoubleComparison,
    RuleWrongValencyCase,
    RuleWrongVerbonominalCase
  )) %>%
  # remove further variables belonging to the 'acceptability' category
  select(!c(RuleIncompleteConjunction)) %>%
  # remove artificially limited variables
  select(!c(
    RuleCaseRepetition.max_repetition_frac,
    RuleCaseRepetition.max_repetition_frac.v
  )) %>%
  # remove variables with too many NAs
  select(!c(
    RuleDoubleAdpos.max_allowable_distance,
    RuleDoubleAdpos.max_allowable_distance.v
  )) %>%
  mutate(across(c(
    class,
    FileFormat,
    subcorpus,
    DocumentVersion,
    LegalActType,
    Objectivity,
    AuthorType,
    RecipientType,
    RecipientIndividuation,
    Anonymized
  ), ~ as.factor(.x)))

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

```
## # A tibble: 0 x 77
## # i 77 variables: KUK_ID <chr>, FileName <chr>, FileFormat <fct>,
## #   subcorpus <fct>, SourceID <chr>, DocumentVersion <fct>,
## #   ParentDocumentID <chr>, LegalActType <fct>, Objectivity <fct>,
## #   Bindingness <lgl>, AuthorType <fct>, RecipientType <fct>,
## #   RecipientIndividuation <fct>, Anonymized <fct>, Recipient Type <chr>,
## #   class <fct>, RuleAbstractNouns <dbl>, RuleAnaphoricReferences <dbl>,
## #   RuleCaseRepetition.max_repetition_count <dbl>, ...
```

```
colnames(data_clean) <- prettify_feat_name_vector(colnames(data_clean))

data_clean_scaled <- data_clean %>%
  mutate(across(class, ~ .x == "good")) %>%
  mutate(across(.firstnonmetacolumn:ncol(data_clean), ~ scale(.x)))
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `across(.firstnonmetacolumn:ncol(data_clean), ~scale(.x))`.
## Caused by warning:
## ! Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(.firstnonmetacolumn)
##
##   # Now:
##   data %>% select(all_of(.firstnonmetacolumn))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

Important features identification

Regularized regression

split the data

```
.no_folds <- 10
.split_prop <- 4 / 5

data_split <- initial_split(data_clean, strata = class, prop = .split_prop)
training_set <- training(data_split)
testing_set <- testing(data_split)

folds <- vfold_cv(training_set, .no_folds)
```

recipe

```
lin_formula <- reformulate(colnames(data_clean)[17:77], "class")
lin_rec <- recipe(lin_formula, data = training_set) %>%
  # step_corr(all_predictors()) %>%
  step_normalize(all_predictors())

lin_wf_base <- workflow() %>% add_recipe(lin_rec)
```

tuning

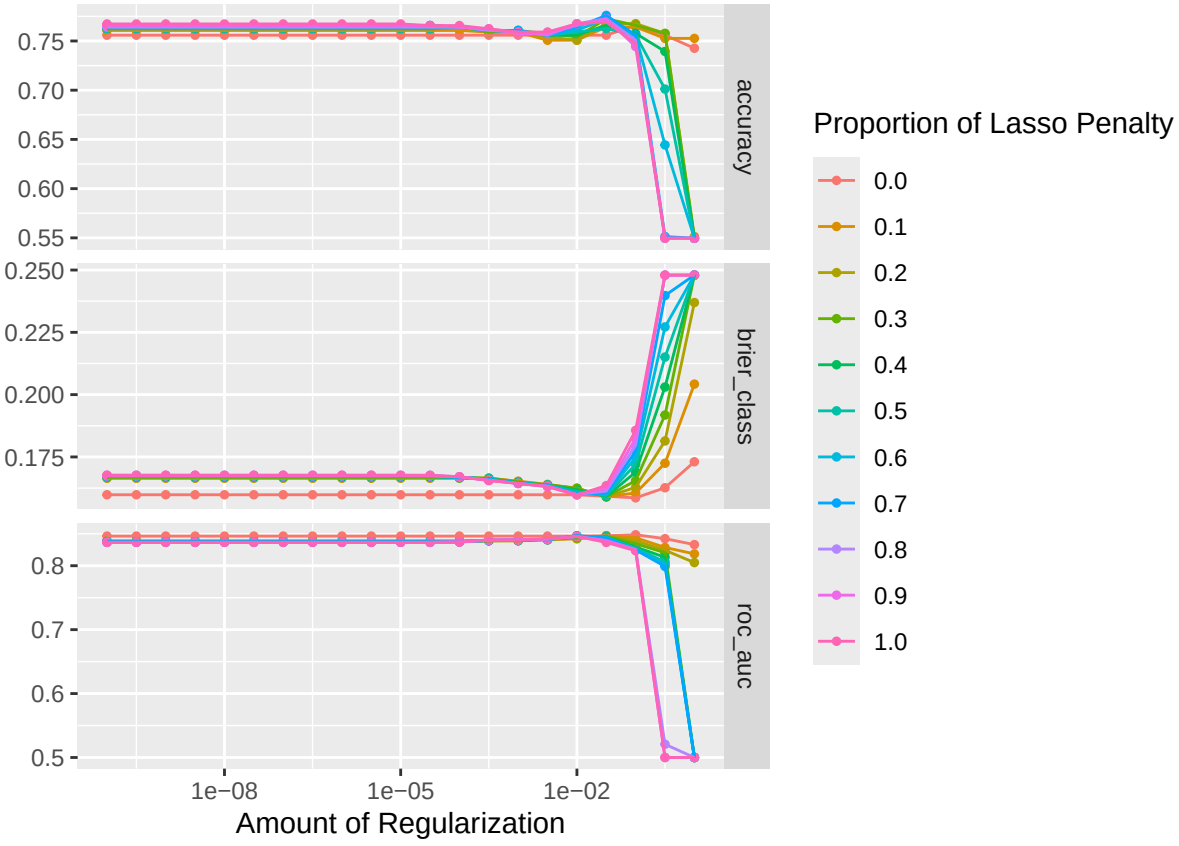
```
lin_wf <- lin_wf_base %>%
  add_model(logistic_reg(
```

```
    mode = "classification", engine = "glmnet",
    penalty = tune(), mixture = tune()
  ))

tune_grid <- grid_regular(
  penalty(), mixture(),
  levels = c(penalty = 21, mixture = 11)
)

tune_rs <- tune_grid(
  lin_wf, folds,
  grid = tune_grid,
  metrics = metric_set(yardstick::accuracy, brier_class, roc_auc)
)

autoplot(tune_rs)
```




```
choose_roc_auc <- tune_rs %>%
  select_by_one_std_err(metric = "roc_auc", -mixture, penalty)
choose_roc_auc
```

```
## # A tibble: 1 x 3
##        penalty mixture .config
##          <dbl>   <dbl> <chr>
## 1 0.0000000001       1 Preprocessor1_Model211
```

final

```
lin_final_wf <- finalize_workflow(lin_wf, choose_roc_auc)
lin_final_wf
```

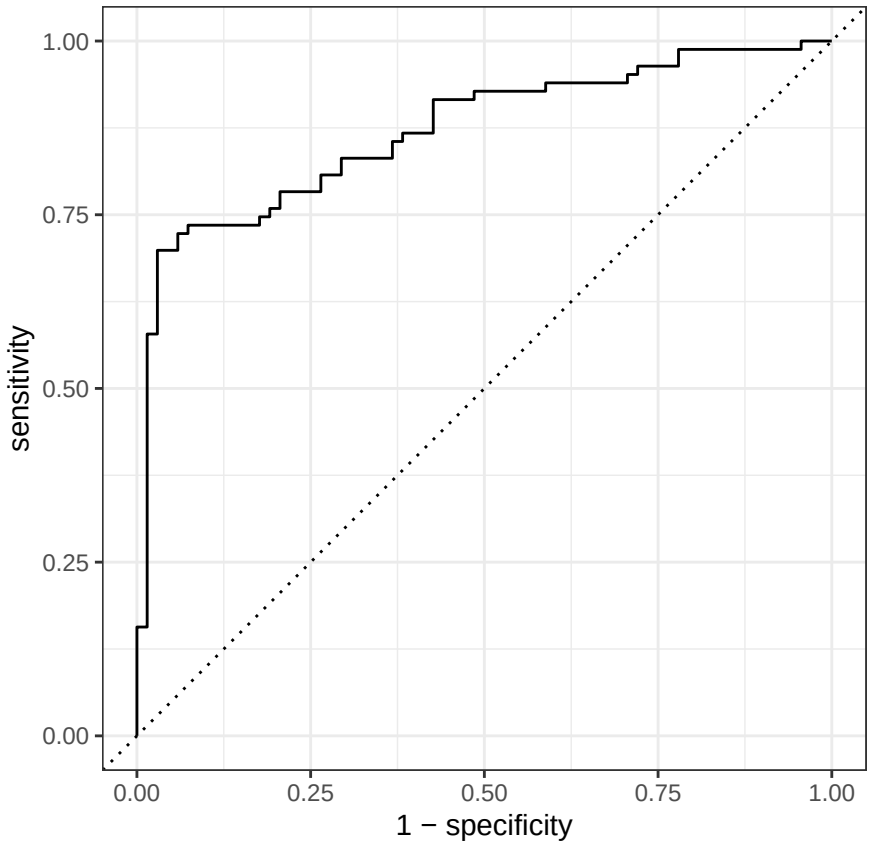
```
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: logistic_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Logistic Regression Model Specification (classification)
##
## Main Arguments:
##   penalty = 1e-10
##   mixture = 1
##
## Computational engine: glmnet
```

```
lin_final_fitted <- last_fit(lin_final_wf, data_split)

collect_predictions(lin_final_fitted) %>%
  conf_mat(truth = class, estimate = .pred_class)
```

```
##           Truth
## Prediction bad good
##       bad   64   14
##       good  19   54
```

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




```
extract_fit_parsnip(lin_final_fitted) %>%
  vip::vi(lambda = choose_roc_auc$penalty) %>%
  print(n = 80)
```

```
## # A tibble: 61 x 3
##    Variable            Importance Sign
##    <chr>                    <dbl> <chr>
##  1 sentlen.m                 2.99 POS
##  2 ari                       2.64 NEG
##  3 gf                        1.96 NEG
##  4 sentcount                 1.86 POS
##  5 atl                       1.41 POS
##  6 activity                  1.37 POS
##  7 VERBfrac.m                1.32 NEG
##  8 smog                      1.17 POS
##  9 hpoint                    1.13 NEG
## 10 wordcount                 1.05 NEG
## 11 ttr                       0.886 NEG
## 12 fre                       0.806 NEG
## 13 entropy.v                 0.720 POS
## 14 entropy                   0.693 NEG
## 15 sentlen.v                 0.580 POS
## 16 ttr.v                     0.541 NEG
## 17 predsubjdist.m            0.493 NEG
## 18 anaphoricrefs             0.447 POS
## 19 cli                       0.430 NEG
## 20 extrcaseexprs             0.411 POS
```

```
## 21 compoundVERBs          0.410    POS
## 22 passives               0.402    NEG
## 23 mattr                  0.347    NEG
## 24 caserepcount.v         0.339    NEG
## 25 predobjdist.m          0.321    NEG
## 26 literary               0.314    NEG
## 27 verbdist               0.308    POS
## 28 caserepcount.m         0.307    POS
## 29 maentropy              0.285    POS
## 30 predorder.m            0.267    NEG
## 31 hapaxes                0.263    POS
## 32 VERBcomp               0.247    POS
## 33 NOUNcount.v            0.227    NEG
## 34 subj                   0.223    POS
## 35 NOUNcount.m            0.212    POS
## 36 VERBcompdist.v         0.208    NEG
## 37 predobjdist.v          0.203    POS
## 38 rfpass_animsubj        0.197    NEG
## 39 NEGcount.m             0.188    POS
## 40 NOUNfrac.m             0.184    NEG
## 41 longexprs              0.179    POS
## 42 redundexprs            0.177    NEG
## 43 compoundVERBsdist.m    0.175    NEG
## 44 doubleADPs             0.168    NEG
## 45 VERBfrac.v             0.157    POS
## 46 relativisticexprs      0.157    NEG
## 47 NEGcount.v             0.145    NEG
## 48 compoundVERBsdist.v    0.139    POS
## 49 NEGfrac.v              0.126    POS
## 50 VERBcompdist.m         0.126    POS
## 51 GPs                    0.105    NEG
## 52 predsubjdist.v         0.0944   NEG
## 53 mamr                   0.0940   NEG
## 54 NOUNfrac.v             0.0857   POS
## 55 obj                    0.0766   POS
## 56 weakmeaning            0.0758   NEG
## 57 predorder.v            0.0467   POS
## 58 verbalNOUNs            0.0348   NEG
## 59 abstractNOUNs          0.00983  POS
## 60 NEGfrac.m              0.000988 POS
## 61 fkgl                   0        NEG
```

```
lin_final_fitted %>%
  extract_fit_parsnip() %>%
  tidy() %>%
  arrange(estimate) %>%
  print(n = 80)
```

```
## # A tibble: 62 x 3
##    term                 estimate      penalty
##    <chr>                   <dbl>        <dbl>
##  1 ari                   -2.64     0.0000000001
##  2 gf                    -1.96     0.0000000001
##  3 VERBfrac.m            -1.32     0.0000000001
##  4 hpoint                -1.13     0.0000000001
```

```
##  5 wordcount           -1.05     0.0000000001
##  6 ttr                 -0.886    0.0000000001
##  7 fre                 -0.806    0.0000000001
##  8 entropy             -0.693    0.0000000001
##  9 (Intercept)         -0.542    0.0000000001
## 10 ttr.v               -0.541    0.0000000001
## 11 predsubjdist.m      -0.493    0.0000000001
## 12 cli                 -0.430    0.0000000001
## 13 passives            -0.402    0.0000000001
## 14 mattr               -0.347    0.0000000001
## 15 caserepcount.v      -0.339    0.0000000001
## 16 predobjdist.m       -0.321    0.0000000001
## 17 literary            -0.314    0.0000000001
## 18 predorder.m         -0.267    0.0000000001
## 19 NOUNcount.v         -0.227    0.0000000001
## 20 VERBcompdist.v      -0.208    0.0000000001
## 21 rfpass_animsubj     -0.197    0.0000000001
## 22 NOUNfrac.m          -0.184    0.0000000001
## 23 redundexprs         -0.177    0.0000000001
## 24 compoundVERBsdist.m -0.175    0.0000000001
## 25 doubleADPs          -0.168    0.0000000001
## 26 relativisticexprs   -0.157    0.0000000001
## 27 NEGcount.v          -0.145    0.0000000001
## 28 GPs                 -0.105    0.0000000001
## 29 predsubjdist.v      -0.0944   0.0000000001
## 30 mamr                -0.0940   0.0000000001
## 31 weakmeaning         -0.0758   0.0000000001
## 32 verbalNOUNs         -0.0348   0.0000000001
## 33 fkgl                 0        0.0000000001
## 34 NEGfrac.m            0.000988 0.0000000001
## 35 abstractNOUNs        0.00983  0.0000000001
## 36 predorder.v          0.0467   0.0000000001
## 37 obj                  0.0766   0.0000000001
## 38 NOUNfrac.v           0.0857   0.0000000001
## 39 VERBcompdist.m       0.126    0.0000000001
## 40 NEGfrac.v            0.126    0.0000000001
## 41 compoundVERBsdist.v  0.139    0.0000000001
## 42 VERBfrac.v           0.157    0.0000000001
## 43 longexprs            0.179    0.0000000001
## 44 NEGcount.m           0.188    0.0000000001
## 45 predobjdist.v        0.203    0.0000000001
## 46 NOUNcount.m          0.212    0.0000000001
## 47 subj                 0.223    0.0000000001
## 48 VERBcomp             0.247    0.0000000001
## 49 hapaxes              0.263    0.0000000001
## 50 maentropy            0.285    0.0000000001
## 51 caserepcount.m       0.307    0.0000000001
## 52 verbdist             0.308    0.0000000001
## 53 compoundVERBs        0.410    0.0000000001
## 54 extrcaseexprs        0.411    0.0000000001
## 55 anaphoricrefs        0.447    0.0000000001
## 56 sentlen.v            0.580    0.0000000001
## 57 entropy.v            0.720    0.0000000001
## 58 smog                 1.17     0.0000000001
```

```
## 59 activity             1.37     0.0000000001
## 60 atl                  1.41     0.0000000001
## 61 sentcount            1.86     0.0000000001
## 62 sentlen.m            2.99     0.0000000001
```

Individual regressions

```
feature_importances <- tibble(
  feat_name = character(),
  p_value = numeric(),
  estimate = numeric(),
  wilcox_p = numeric(),
  wilcox_r = numeric(),
  kw_p = numeric(),
  kw_epsilon2 = numeric()
)

for (i in .firstnonmetacolumn:ncol(data_clean)) {
  fname <- names(data_clean)[i]

  formula_single <- reformulate(fname, "class")
  formula_single_reversed <- reformulate("class", fname)

  glm_model <- glm(formula_single, data_clean, family = "binomial")
  glm_coefficients <- summary(glm_model)$coefficients
  row_index <- which(rownames(glm_coefficients) == fname)
  p_value <- glm_coefficients[row_index, 4]
  beta <- glm_coefficients[row_index, 1]

  wilcox_p <- wilcox.test(formula_single_reversed, data_clean)$p.value
  wilcox_r <- wilcox_effsize(data_clean, formula_single_reversed)$effsize[[1]]

  kw_p <- kruskal.test(data_clean[[fname]], data_clean$class)$p.value
  kw_epsilon2 <- epsilonSquared(data_clean[[fname]], data_clean$class)[[1]]

  feature_importances <- feature_importances %>%
    add_row(
      feat_name = fname,
      p_value = p_value,
      estimate = beta,
      wilcox_p = wilcox_p,
      wilcox_r = wilcox_r,
      kw_p = kw_p,
      kw_epsilon2 = kw_epsilon2
    )
}
feature_importances
```

```
## # A tibble: 61 x 7
##    feat_name       p_value  estimate wilcox_p wilcox_r     kw_p kw_epsilon2
##    <chr>             <dbl>     <dbl>    <dbl>    <dbl>    <dbl>       <dbl>
##  1 abstractNOUNs  2.20e- 3   85.5    6.39e- 3   0.0994 6.39e- 3     0.00989
##  2 anaphoricrefs  6.73e- 1   34.3    9.80e- 3   0.0941 9.79e- 3     0.00887
##  3 caserepcount.m 6.59e- 2   -1.02   7.61e- 2   0.0647 7.60e- 2     0.00419
```

```
##  4 caserepcount.v 4.54e- 3   -2.08   9.43e- 4   0.121  9.43e- 4     0.0145
##  5 extrcaseexprs  1.08e- 1 -347.     1.34e- 3   0.117  1.34e- 3     0.0137
##  6 doubleADPs     2.71e- 1  -24.8    3.02e- 1   0.0376 3.02e- 1     0.00141
##  7 VERBcomp       5.24e-15    4.89   1.36e-16   0.301  1.36e-16     0.0909
##  8 VERBcompdist.m 5.48e- 2   -0.0900 1.73e- 2   0.0868 1.73e- 2     0.00754
##  9 VERBcompdist.v 6.58e- 2   -0.327  7.90e- 2   0.0640 7.89e- 2     0.0041
## 10 literary       7.00e-21 -245.     1.44e-26   0.389  1.44e-26     0.151
## # i 51 more rows
```

```
selected_features <- feature_importances %>%
  mutate(
    selected = p_value <= 0.05,
    wilcox_sel = wilcox_p < 0.05,
    kw_sel = kw_p < 0.05
  )

selected_features %>%
  select(selected, kw_sel) %>%
  table()
```

```
##         kw_sel
## selected FALSE TRUE
##    FALSE     8    4
##    TRUE      4   45
```

```
cor(-log(selected_features$p_value), selected_features$kw_epsilon2)
```

```
## [1] 0.952316
```

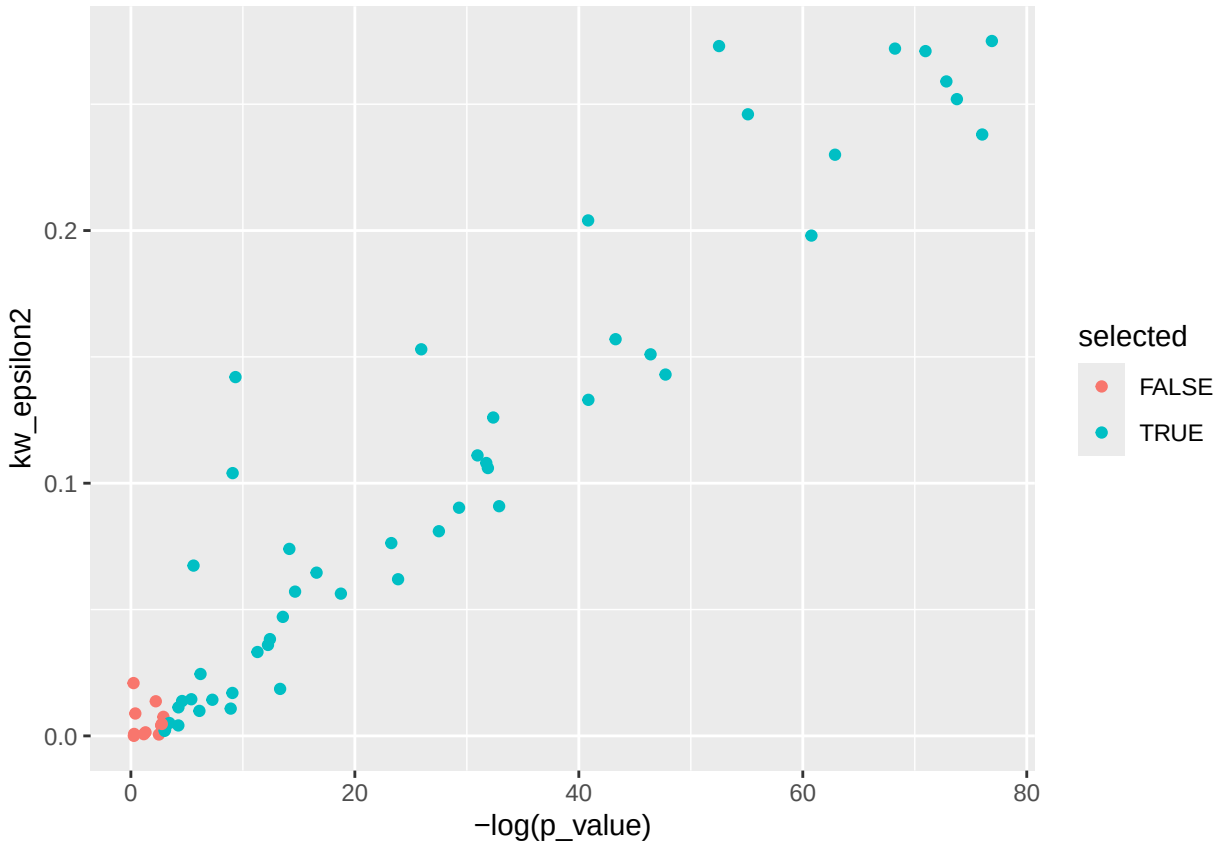
```
cor(-log(selected_features$p_value), -log(selected_features$kw_p))
```

```
## [1] 0.9524106
```

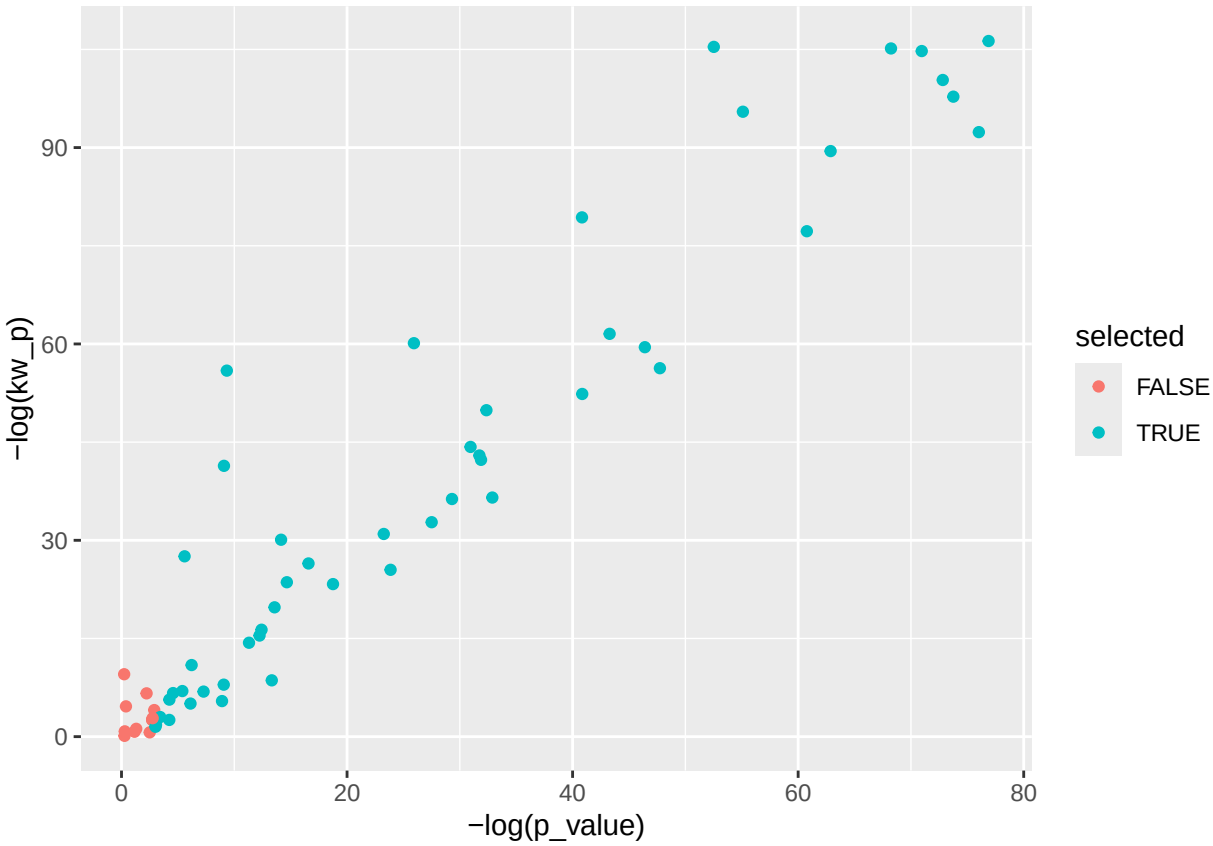
```
cor(selected_features$estimate, selected_features$kw_epsilon2)
```

```
## [1] 0.1146951
```

```
selected_features %>%
  ggplot(aes(x = -log(p_value), y = kw_epsilon2, color = selected)) +
  geom_point()
```




```
selected_features %>%
  ggplot(aes(x = -log(p_value), y = -log(kw_p), color = selected)) +
  geom_point()
```




```
selected_features_names <- selected_features %>%
  filter(selected) %>%
  pull(feat_name)
```

Compare the two

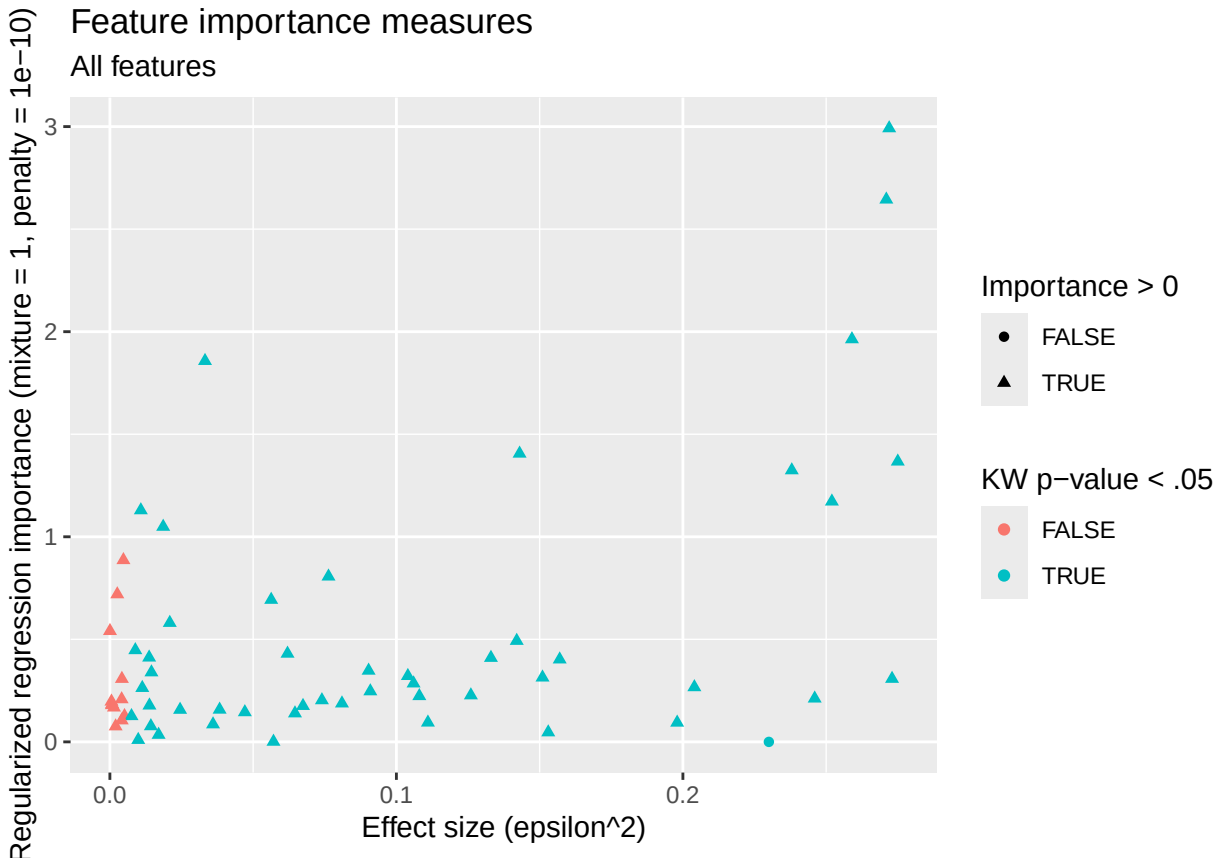
```
featcomp <- extract_fit_parsnip(lin_final_fitted) %>%
  vip::vi(lambda = choose_roc_auc$penalty) %>%
  full_join(
    selected_features %>% rename(Variable = feat_name),
    by = "Variable"
  ) %>%
  rename(selected_pval = selected) %>%
  mutate(
    log_p = -log(p_value),
    log_wilcox_p = -log(wilcox_p),
    log_kw_p = -log(kw_p),
    selected_reg = Importance > 0
  )

featcomp %>%
  filter(!is.na(Importance)) %>%
  select(Importance, kw_epsilon2, log_p, log_kw_p) %>%
  cor() %>%
  round(2)
```

```
##             Importance kw_epsilon2 log_p log_kw_p
```

```
## Importance        1.00        0.47  0.51     0.47
## kw_epsilon2       0.47        1.00  0.95     1.00
## log_p             0.51        0.95  1.00     0.95
## log_kw_p          0.47        1.00  0.95     1.00
```

```r
featcomp_plot <- featcomp %>% ggplot(aes(
  x = kw_epsilon2,
  y = Importance,
  # size = log_p,
  color = kw_sel,
  shape = selected_reg
)) +
  geom_point() +
  labs(
    title = "Feature importance measures",
    subtitle = "All features",
    # subtitle = "Features with |r| < 0.90",
    x = "Effect size (epsilon^2)",
    y = paste0(c(
      "Regularized regression importance (mixture = ",
      choose_roc_auc$mixture[1], ", penalty = ",
      choose_roc_auc$penalty[1], ")"
    ), collapse = ""),
    # size = "-log(p-value)",
    color = "KW p-value < .05",
    shape = "Importance > 0"
  )
print(featcomp_plot)
```




```
ggsave("featcomp_all.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
# ggsave("featcomp_nocorr.png")
```