Resampling with TidyModels

clinton moshe

11/18/2021

The Libraries

```
library(tidymodels)
```

```
## Registered S3 method overwritten by 'tune':
##   method                   from
##   required_pkgs.model_spec parsnip
```

```
## -- Attaching packages -------------------------------------- tidymodels 0.1.4 --
```

```
## v broom        0.7.10     v recipes      0.1.17
## v dials        0.0.10     v rsample      0.1.1
## v dplyr        1.0.7      v tibble       3.1.6
## v ggplot2      3.3.5      v tidyr        1.1.4
## v infer        1.0.0      v tune         0.1.6
## v modeldata    0.1.1      v workflows    0.2.4
## v parsnip      0.1.7      v workflowsets 0.1.0
## v purrr        0.3.4      v yardstick    0.0.8
```

```
## Warning: package 'dials' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'infer' was built under R version 4.0.5
```

```
## Warning: package 'recipes' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'tune' was built under R version 4.0.5
```

```
## Warning: package 'workflows' was built under R version 4.0.5
```

```
## Warning: package 'workflowsets' was built under R version 4.0.5
```

```
## Warning: package 'yardstick' was built under R version 4.0.5

## -- Conflicts ---------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Dig deeper into tidy modeling with R at https://www.tmwr.org
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v readr   2.1.0     v forcats 0.5.1
## v stringr 1.4.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## x dplyr::filter()     masks stats::filter()
## x stringr::fixed()    masks recipes::fixed()
## x dplyr::lag()        masks stats::lag()
## x readr::spec()       masks yardstick::spec()
```

The Data

```
diabetes <- read_csv('data/diabetes.csv')
```

```
## Rows: 768 Columns: 9

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (9): Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, D...

##
## 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.
```

```
# structure of the data
str(diabetes)
```

```
## spec_tbl_df [768 x 9] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Pregnancies             : num [1:768] 6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : num [1:768] 148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : num [1:768] 72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : num [1:768] 35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : num [1:768] 0 0 0 94 168 0 88 0 543 0 ...
```

```
##  $ BMI                     : num [1:768] 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ DiabetesPedigreeFunction: num [1:768] 0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : num [1:768] 50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : num [1:768] 1 0 1 0 1 0 1 0 1 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Pregnancies = col_double(),
##   ..   Glucose = col_double(),
##   ..   BloodPressure = col_double(),
##   ..   SkinThickness = col_double(),
##   ..   Insulin = col_double(),
##   ..   BMI = col_double(),
##   ..   DiabetesPedigreeFunction = col_double(),
##   ..   Age = col_double(),
##   ..   Outcome = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
# summary stat
summary(diabetes)
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##     Outcome
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
##  Max.   :1.000
```

```
# convert outcome to factor
diabetes <- diabetes %>%
  mutate(Outcome = as_factor(Outcome))
```

Data Visualization

You can do your EDA here

Data Splitting

```
data_split <- initial_split(diabetes, prop = 7/10)

# train
train <- training(data_split)

# test
test <- testing(data_split)
```

Recipe

```
rf_rec <-
  recipe(Outcome~., data = train) %>%
  step_normalize(all_predictors())

rf_rec
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          8
##
## Operations:
##
## Centering and scaling for all_predictors()
```

Model

```
rf_mod <-
  rand_forest(trees = 300) %>%
  set_engine('ranger') %>%
  set_mode('classification')

rf_mod
```

```
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   trees = 300
##
## Computational engine: ranger
```

Workflow

```
rf_wf <-
  workflow() %>%
  add_model(rf_mod) %>%
  add_recipe(rf_rec)
rf_wf
```

```
## == Workflow ====================================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Random Forest Model Specification (classification)
##
## Main Arguments:
##   trees = 300
##
## Computational engine: ranger
```

Model Fitting

```
rf_fit <-
  rf_wf %>%
  fit(train)
rf_fit
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_normalize()
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, num.trees = ~300,      num.threads = 1, verbose = FA
##
## Type:                             Probability estimation
## Number of trees:                  300
## Sample size:                      537
```

```
## Number of independent variables:  8
## Mtry:                             2
## Target node size:                 10
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.1615517
```

Model Performance

train

```
# predict on the train data
rf_train_pred <-
  predict(rf_fit, new_data = train) %>%
  bind_cols(predict(rf_fit, new_data = train, type = 'prob')) %>% # combining predicted with actual val
  bind_cols(train %>% select(Outcome))
rf_train_pred
```

```
## # A tibble: 537 x 4
##    .pred_class .pred_0 .pred_1 Outcome
##    <fct>         <dbl>   <dbl> <fct>
##  1 0             0.650 0.350   0
##  2 0             0.838 0.162   0
##  3 0             0.997 0.00333 0
##  4 0             0.831 0.169   0
##  5 0             0.950 0.0505  0
##  6 0             0.735 0.265   0
##  7 1             0.346 0.654   1
##  8 1             0.143 0.857   1
##  9 1             0.218 0.782   1
## 10 1             0.136 0.864   1
## # ... with 527 more rows
```

test

```
# predict on the test data and compare results
rf_test_pred <-
  predict(rf_fit, new_data = test) %>%
  bind_cols(predict(rf_fit, new_data = test, type = 'prob')) %>% # combining predicted with actual valu
  bind_cols(test %>% select(Outcome))
rf_test_pred
```

```
## # A tibble: 231 x 4
##    .pred_class .pred_0 .pred_1 Outcome
##    <fct>         <dbl>   <dbl> <fct>
##  1 1             0.402   0.598 1
##  2 1             0.342   0.658 1
##  3 1             0.447   0.553 0
##  4 0             0.804   0.196 0
```

```
##  5 1             0.202   0.798 1
##  6 1             0.319   0.681 1
##  7 0             0.546   0.454 1
##  8 0             0.704   0.296 0
##  9 0             0.565   0.435 0
## 10 0             0.578   0.422 1
## # ... with 221 more rows
```

Predictions Accuracy

```
# train acc
rf_train_pred %>%
  accuracy(Outcome, .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.957
```

```
# test acc
rf_test_pred %>%
  accuracy(Outcome, .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.771
```

Resampling

```
# defining the folds for cross vallidation
set.seed(100)

folds <- vfold_cv(train,10,repeats = 2)

# fitting the model
set.seed(102)
rf_fit_cv <-
  rf_wf %>%
  fit_resamples(folds)
```

```
## Warning: package 'rlang' was built under R version 4.0.5
```

```
## Warning: package 'vctrs' was built under R version 4.0.5
```

```
rf_fit_cv
```

```
## # Resampling results
## # 10-fold cross-validation repeated 2 times
## # A tibble: 20 x 5
##    splits           id      id2    .metrics         .notes
##    <list>           <chr>   <chr>  <list>           <list>
##  1 <split [483/54]> Repeat1 Fold01 <tibble [2 x 4]> <tibble [0 x 1]>
##  2 <split [483/54]> Repeat1 Fold02 <tibble [2 x 4]> <tibble [0 x 1]>
##  3 <split [483/54]> Repeat1 Fold03 <tibble [2 x 4]> <tibble [0 x 1]>
##  4 <split [483/54]> Repeat1 Fold04 <tibble [2 x 4]> <tibble [0 x 1]>
##  5 <split [483/54]> Repeat1 Fold05 <tibble [2 x 4]> <tibble [0 x 1]>
##  6 <split [483/54]> Repeat1 Fold06 <tibble [2 x 4]> <tibble [0 x 1]>
##  7 <split [483/54]> Repeat1 Fold07 <tibble [2 x 4]> <tibble [0 x 1]>
##  8 <split [484/53]> Repeat1 Fold08 <tibble [2 x 4]> <tibble [0 x 1]>
##  9 <split [484/53]> Repeat1 Fold09 <tibble [2 x 4]> <tibble [0 x 1]>
## 10 <split [484/53]> Repeat1 Fold10 <tibble [2 x 4]> <tibble [0 x 1]>
## 11 <split [483/54]> Repeat2 Fold01 <tibble [2 x 4]> <tibble [0 x 1]>
## 12 <split [483/54]> Repeat2 Fold02 <tibble [2 x 4]> <tibble [0 x 1]>
## 13 <split [483/54]> Repeat2 Fold03 <tibble [2 x 4]> <tibble [0 x 1]>
## 14 <split [483/54]> Repeat2 Fold04 <tibble [2 x 4]> <tibble [0 x 1]>
## 15 <split [483/54]> Repeat2 Fold05 <tibble [2 x 4]> <tibble [0 x 1]>
## 16 <split [483/54]> Repeat2 Fold06 <tibble [2 x 4]> <tibble [0 x 1]>
## 17 <split [483/54]> Repeat2 Fold07 <tibble [2 x 4]> <tibble [0 x 1]>
## 18 <split [484/53]> Repeat2 Fold08 <tibble [2 x 4]> <tibble [0 x 1]>
## 19 <split [484/53]> Repeat2 Fold09 <tibble [2 x 4]> <tibble [0 x 1]>
## 20 <split [484/53]> Repeat2 Fold10 <tibble [2 x 4]> <tibble [0 x 1]>
```

Samples metrics

```
rf_fit_cv %>%
  collect_metrics()
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.757    20  0.0117 Preprocessor1_Model1
## 2 roc_auc  binary     0.817    20  0.0106 Preprocessor1_Model1
```