AE 10: Cross validation

The Office

Oct 24, 2022

> ❗ Important
>
> This AE is due on GitHub by Thursday, October 27 at 11:59pm.

Function

```
# function to calculate model fit statistics
calc_model_stats <- function(x) {
  glance(extract_fit_parsnip(x)) |>
    select(adj.r.squared, AIC, BIC)
}
```

Packages

```
library(tidyverse)
library(tidymodels)
library(knitr)
```

Load data

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

Split data into training and testing

Split your data into testing and training sets.

```
set.seed(123)
office_split <- initial_split(office_episodes)
office_train <- training(office_split)
office_test <- testing(office_split)
```

Specify model

Specify a linear regression model. Call it `office_spec`.

```
office_spec <- linear_reg() |>
  set_engine("lm")

office_spec
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Model 1

Create recipe

Create the recipe from class. Call it `office_rec1`.

```
office_rec1 <- recipe(imdb_rating ~ ., data = office_train) |>
  update_role(episode_name, new_role = "id") |>
  step_rm(air_date, season) |>
  step_dummy(all_nominal_predictors()) |>
  step_zv(all_predictors())

office_rec1
```

```
Recipe
```

```
Inputs:

      role #variables
        id          1
   outcome          1
 predictor         12

Operations:

Delete terms air_date, season
Dummy variables from all_nominal_predictors()
Zero variance filter on all_predictors()
```

Preview recipe

```
prep(office_rec1) |>
  bake(office_train) |>
  glimpse()
```

```
Rows: 139
Columns: 12
$ episode       <dbl> 20, 16, 8, 7, 23, 3, 16, 21, 18, 14, 27, 28, 12, 1, 23, ~
$ episode_name  <fct> "Welcome Party", "Moving On", "Performance Review", "The~
$ total_votes   <dbl> 1489, 1572, 2416, 1406, 2783, 1802, 2283, 2041, 1445, 14~
$ lines_jim     <dbl> 0.12703583, 0.05588822, 0.09523810, 0.07482993, 0.078291~
$ lines_pam     <dbl> 0.10423453, 0.10978044, 0.10989011, 0.15306122, 0.081850~
$ lines_michael <dbl> 0.0000000, 0.0000000, 0.3772894, 0.0000000, 0.3736655, 0~
$ lines_dwight  <dbl> 0.07166124, 0.08782435, 0.15384615, 0.18027211, 0.135231~
$ imdb_rating   <dbl> 7.2, 8.2, 8.2, 7.7, 9.1, 8.2, 8.3, 8.9, 8.0, 7.8, 8.7, 8~
$ halloween_yes <dbl> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
$ valentine_yes <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
$ christmas_yes <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0,~
$ michael_yes   <dbl> 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1,~
```

Create workflow

Create the workflow that brings together the model specification and recipe. Call it `office_wflow1`.

```r
office_wflow1 <- workflow() |>
  add_model(office_spec) |>
  add_recipe(office_rec1)

office_wflow1
```

```
== Workflow ====================================================================
Preprocessor: Recipe
Model: linear_reg()

-- Preprocessor ----------------------------------------------------------------
3 Recipe Steps

* step_rm()
* step_dummy()
* step_zv()

-- Model -----------------------------------------------------------------------
Linear Regression Model Specification (regression)

Computational engine: lm
```

Cross validation

Create folds

Create 10-folds.

```r
# make 10 folds
set.seed(345)
folds <- vfold_cv(office_train, v = 10)
```

Conduct cross validation

Conduct cross validation on the 10 folds.

```r
set.seed(456)
# Fit model and calculate statistics for each fold
office_fit_rs1 <- office_wflow1 |>
```

```
  fit_resamples(resamples = folds,
                control = control_resamples(extract = calc_model_stats))
```

```
Warning: package 'rlang' was built under R version 4.1.3
```

Summarize assessment CV metrics

Summarize assessment metrics from your CV resamples.

```
collect_metrics(office_fit_rs1, summarize = TRUE)
```

```
# A tibble: 2 x 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 rmse    standard   0.339    10  0.0227 Preprocessor1_Model1
2 rsq     standard   0.567    10  0.0458 Preprocessor1_Model1
```

Summarize model fit CV metrics

```
map_df(office_fit_rs1$.extracts, ~ .x[[1]][[1]]) |>
  summarise(mean_adj_rsq = mean(adj.r.squared),
            mean_aic = mean(AIC),
            mean_bic = mean(BIC))
```

```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.583     90.8     125.
```

Another model - Model 2

Create a different (simpler, involving fewer variables) recipe and call it `office_rec2`. Conduct 10-fold cross validation and summarize metrics.

Model 2: Recipe

```
office_rec2 <- recipe(imdb_rating ~ season + episode + air_date + michael, data = office_t
  step_date(air_date)
```

Model 2: Model building workflow

```
office_wflow2 <- workflow() %>%
  add_model(office_spec) %>%
  add_recipe(office_rec2)
```

Model 2: Conduct CV

> Note
>
> Note: We will use the same folds as the ones used for Model 1. Why should we use the same folds to evaluate and compare both models?

```
set.seed(345)
office_fit_rs2 <- office_wflow2 |>
  fit_resamples(resamples = folds,
                control = control_resamples(extract = calc_model_stats))
```

```
! Fold01: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold02: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold03: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold04: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold05: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold06: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold07: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
! Fold08: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
! Fold09: preprocessor 1/1, model 1/1 (predictions): prediction from a rank-defici...
```

```
x Fold10: preprocessor 1/1, model 1/1 (predictions): Error in model.frame.default(...
```

Model 2: Summarize assessment CV metrics

```
collect_metrics(office_fit_rs2, summarize = T)
```

```
# A tibble: 2 x 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 rmse    standard   0.432     9  0.0374 Preprocessor1_Model1
2 rsq     standard   0.277     9  0.0729 Preprocessor1_Model1
```

Model 2: Summarize model fit CV metrics

```
map_df(office_fit_rs2$.extracts, ~ .x[[1]][[1]]) |>
  summarise(mean_adj_rsq = mean(adj.r.squared),
            mean_aic = mean(AIC),
            mean_bic = mean(BIC))
```

```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.315     156.     201.
```

Compare models

The model I created performs worse on the training data than the model created above. It had a higher RMSE value (0.432 vs 0.339) and lower R^2 value (0.277 vs 0.567) than the original model, suggesting that it has a higher error on average and accounts for less of the variability in imdb_data. Additionally, when we look at more advanced model comparison metrics, it again performs worse. Looking at adjusted R^2, which calculates the amount of the variance in the response variable that can be explained by the predictors while penalizing the model for having too many predictors, even though my model has fewer predictors it still performs far worse (0.315 vs 0.583). And, it has much higher AIC and BIC values, which we generally want to lower. Thus, according to this analysis, my model performs worse than the class model.

Use RMSE and BIC for prediction, Adj R^2 and AIC for explanation.

Submission

! Important

To submit the AE:

- Render the document to produce the PDF with all of your work from today's class.
- Push all your work to your `ae-10-` repo on GitHub. (You do not submit AEs on Gradescope).