AE 08: Feature Engineering

The Office

Oct 05, 2022

> **!** Important
>
> The AE is due on GitHub by Saturday, October 08 at 11:59pm.

Packages

```r
library(tidyverse)
library(tidymodels)
library(viridis)
library(knitr)
```

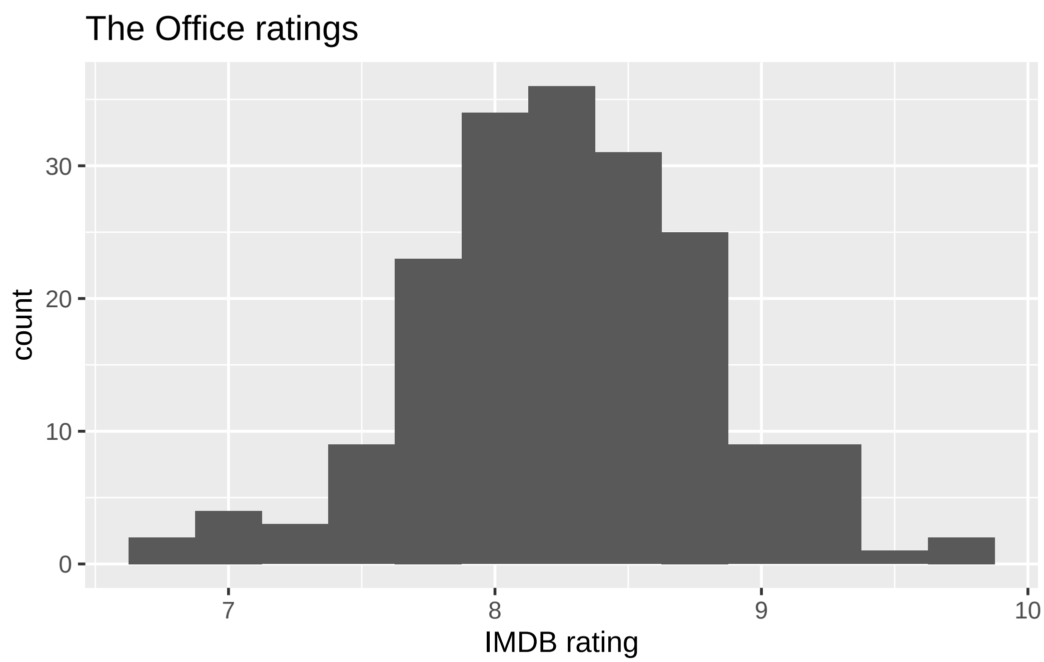
Load data

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

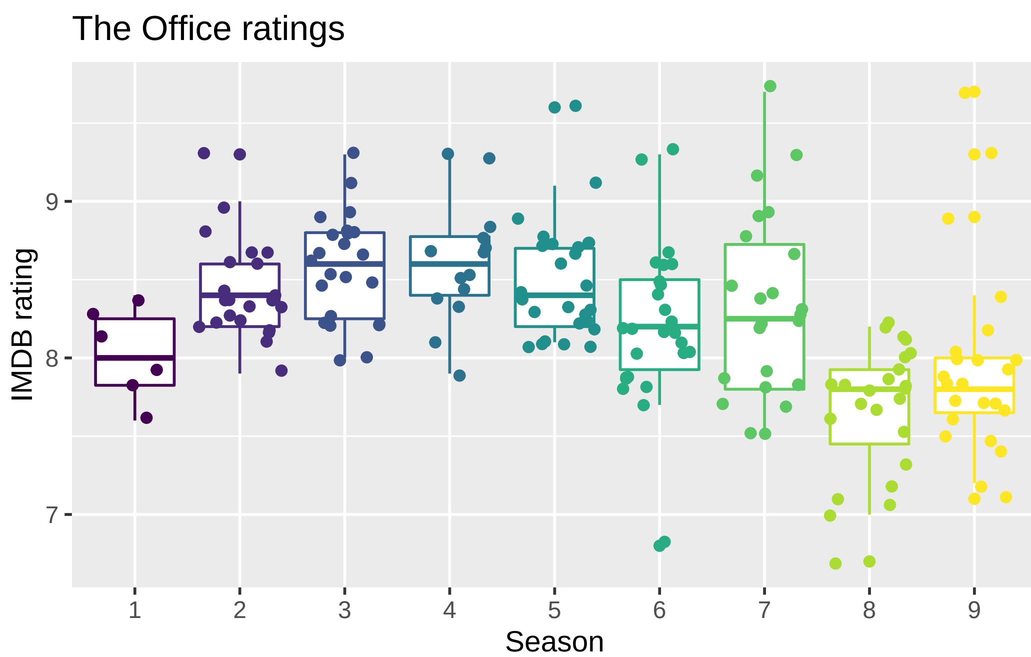
Exploratory data analysis

Below are two of the exploratory data analysis plots from lecture.

```r
ggplot(office_ratings, aes(x = imdb_rating)) +
  geom_histogram(binwidth = 0.25) +
  labs(
    title = "The Office ratings",
    x = "IMDB rating"
  )
```




```
office_ratings |>
  mutate(season = as_factor(season)) |>
  ggplot(aes(x = season, y = imdb_rating, color = season)) +
  geom_boxplot() +
  geom_jitter() +
  guides(color = "none") +
  labs(
    title = "The Office ratings",
    x = "Season",
    y = "IMDB rating"
  ) +
  scale_color_viridis_d()
```



Test/train split

```r
set.seed(123)
office_split <- initial_split(office_ratings) # prop = 3/4 by default
office_train <- training(office_split)
office_test  <- testing(office_split)
```

Build a recipe

```r
office_rec <- recipe(imdb_rating ~ ., data = office_train) |>
  # make title's role ID
  update_role(title, new_role = "ID") |>
  # extract day of week and month of air_date
  step_date(air_date, features = c("dow", "month")) |>
  # identify holidays and add indicators
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay",
                 "USNewYearsDay", "USIndependenceDay"),
```

```r
    keep_original_cols = FALSE
  ) |>
  # turn season into factor
  step_num2factor(season, levels = as.character(1:9)) |>
  # make dummy variables
  step_dummy(all_nominal_predictors()) |>
  # remove zero variance predictors
  step_zv(all_predictors())
```

```r
office_rec
```

```
Recipe

Inputs:

      role #variables
        ID          1
   outcome          1
 predictor          4

Operations:

Date features from air_date
Holiday features from air_date
Factor variables from season
Dummy variables from all_nominal_predictors()
Zero variance filter on all_predictors()
```

Workflows and model fitting

Specify model

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

office_spec
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Build workflow

```r
office_wflow <- workflow() |>
  add_model(office_spec) |>
  add_recipe(office_rec)
```

```r
office_wflow
```

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

-- Preprocessor ----------------------------------------------------------------
5 Recipe Steps

* step_date()
* step_holiday()
* step_num2factor()
* step_dummy()
* step_zv()

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

Computational engine: lm
```

Fit model to training data

```r
office_fit <- office_wflow |>
  fit(data = office_train)

tidy(office_fit) |>
  kable(digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	6.396	0.510	12.532	0.000
episode	-0.004	0.017	-0.230	0.818
total_votes	0.000	0.000	9.074	0.000
season_X2	0.811	0.327	2.482	0.014

term	estimate	std.error	statistic	p.value
season_X3	1.042	0.343	3.040	0.003
season_X4	1.090	0.295	3.695	0.000
season_X5	1.082	0.348	3.109	0.002
season_X6	1.004	0.367	2.735	0.007
season_X7	1.018	0.352	2.894	0.005
season_X8	0.497	0.348	1.430	0.155
season_X9	0.621	0.345	1.802	0.074
air_date_dow_Tue	0.382	0.422	0.904	0.368
air_date_dow_Thu	0.284	0.389	0.731	0.466
air_date_month_Feb	-0.060	0.132	-0.452	0.652
air_date_month_Mar	-0.075	0.156	-0.481	0.631
air_date_month_Apr	0.095	0.177	0.539	0.591
air_date_month_May	0.156	0.213	0.734	0.464
air_date_month_Sep	-0.078	0.223	-0.348	0.728
air_date_month_Oct	-0.176	0.174	-1.014	0.313
air_date_month_Nov	-0.156	0.149	-1.046	0.298
air_date_month_Dec	0.170	0.149	1.143	0.255

Evaluate model on training data

Make predictions

> **!** Important
>
> Fill in the code and make `#| eval: true` before rendering the document.

```
office_train_pred <- predict(office_fit, office_train) |>
  bind_cols(office_train)

office_train_pred
```

```
# A tibble: 141 x 7
   .pred season episode title                  imdb_rating total_votes air_date
   <dbl>  <dbl>   <dbl> <chr>                        <dbl>       <dbl> <date>
 1  7.57      8      18 Last Day in Florida            7.8        1429 2012-03-08
 2  7.77      9      14 Vandalism                      7.6        1402 2013-01-31
 3  8.31      2       8 Performance Review             8.2        2416 2005-11-15
 4  7.67      9       5 Here Comes Treble              7.1        1515 2012-10-25
```

```
 5  8.84       3       22 Beach Games                 9.1         2783 2007-05-10
 6  8.33       7        1 Nepotism                    8.4         1897 2010-09-23
 7  8.46       3       15 Phyllis' Wedding            8.3         2283 2007-02-08
 8  8.14       9       21 Livin' the Dream            8.9         2041 2013-05-02
 9  7.87       9       18 Promos                      8           1445 2013-04-04
10  7.74       8       12 Pool Party                  8           1612 2012-01-19
# ... with 131 more rows
```

Calculate R^2

> **!** Important
>
> Fill in the code and make `#| eval: true` before rendering the document.

```r
rsq(office_train_pred, truth = imdb_rating, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rsq     standard       0.670
```

- What is preferred - high or low values of R^2?

We prefer to have high values of R^2, because those indicate that our model explains a larger amount of the variance in our response variable.

Calculate RMSE

> **!** Important
>
> Fill in the code and make `#| eval: true` before rendering the document.

```r
rmse(office_train_pred, truth = imdb_rating, estimate = .pred)
```

- What is preferred - high or low values of RMSE?

We prefer to have low values of RMSE, because this corresponds with lower error in our model.

- Is this RMSE considered high or low? *Hint: Consider the range of the response variable to answer this question.*

This is probably considered low RMSE - it only reflects about 10% of the range in the response variable, meaning our prediction is generally pretty accurate.

````
::: {.cell}

```{.r .cell-code}
office_train |>
  summarise(min = min(imdb_rating), max = max(imdb_rating))
```

::: {.cell-output .cell-output-stdout}
```
# A tibble: 1 x 2
    min   max
  <dbl> <dbl>
1   6.7   9.7
```
:::
:::
````

Evaluate model on testing data

Answer the following before evaluating the model performance on testing data:

- Do you expect R^2 on the testing data to be higher or lower than the R^2 calculated using training data? Why?

Lower - the model was built to explain variance in the existing data, and it would be somewhat surprising if it were better at explaining variance in a completely different set of data on which it was not trained.

- Do you expect RMSE on the testing data to be higher or lower than the R^2 calculated using training data? Why?

Higher - the model was built to minimize error in the prediction of IMDB ratings based on the training data we had. It would be surprising if it had even less error when predicting IMDB ratings in other data.

Make predictions

```
# fill in code to make predictions from testing data
testPred <- predict(office_fit, office_test) %>%
  bind_cols(office_test)
```

Calculate R^2

```
# fill in code to calculate $R^2$ for testing data
rsq(testPred, truth = imdb_rating, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rsq     standard       0.468
```

Calculate RMSE

```
# fill in code to calculate RMSE for testing data
rmse(testPred, truth = imdb_rating, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard       0.411
```

Compare training and testing data results

- Compare the R^2 for the training and testing data. Is this what you expected?

The R^2 of the testing data is notably lower than that of the training data. This is exactly what we would expect, because the model was not built on the testing data and thus cannot provide as accurate of a prediction.

- Compare the RMSE for the training and testing data. Is this what you expected?

The RMSE of the testing data is notably higher than that of the training data. This is, again, exactly what we would expect - the model was not built on the training data, so it will provide predictions for the IMDB ratings of episodes in the testing data that have more error.

! 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-08-` repo on GitHub. (You do not submit AEs on Gradescope).