AE 14: Logistic regression

Model comparsion

Nov 14, 2022

! Important

The AE is due on GitHub by Thursday, November 17, 11:59pm.

i Note

As you work on the AE, post questions on Ed Discussion. We will also use this space to get responses from groups:

- 10:15am lecture: https://edstem.org/us/courses/26900/discussion/2154489
- 3:30pm lecture : https://edstem.org/us/courses/26900/discussion/2154491

As you work on the AE, post questions on Ed Discussion. We will also use this space to get responses from groups:

- 10:15am lecture: https://edstem.org/us/courses/26900/discussion/2154489
- 3:30pm lecture : https://edstem.org/us/courses/26900/discussion/2154491

Packages

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

Data

For this application exercise we will work with a data set of 25,000 randomly sampled flights that departed one of three NYC airports (JFK, LGA, EWR) in 2013.

```
flight_data <- read_csv("data/flight-data.csv")
```

The goal of this analysis is to fit a model that could be used to predict whether a flight will arrive on time (up to 30 minutes past the scheduled arrival time) or late (more than 30 minutes past the scheduled arrival time).

1. Convert `arr_delay` to factor with levels `"late"` (first level) and `"on_time"` (second level). This variable is our outcome and it indicates whether the flight's arrival was more than 30 minutes.

```
flight_data <- flight_data %>%
  mutate(arr_delay = factor(arr_delay, levels = c("late", "on_time")))
```

Modeling prep

2. Split the data into testing (75%) and training (25%), and save each subset.

```
set.seed(222)
flightSplit <- initial_split(flight_data, prop = 0.5)
train <- training(flightSplit)
test <- testing(flightSplit)
```

3. Specify a logistic regression model that uses the `"glm"` engine.

```
spec <- logistic_reg() %>%
  set_engine("glm")
```

Next, we'll create two recipes and workflows and compare them to each other.

Model 1: Everything and the kitchen sink (12 minutes)

4. Define a recipe that predicts `arr_delay` using all variables except for `flight` and `time_hour`, which, in combination, can be used to identify a flight, and the variable `dest`. Also make sure this recipe handles dummy coding as well as issues that can arise due to having categorical variables with some levels apparent in the training set but not in the testing set. Call this recipe `flights_rec1`.

```r
flights_rec1 <- recipe(arr_delay ~ dep_time + origin + air_time + distance + carrier + dat
  #step_other(all_nominal_predictors()) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors())
```

5. Create a workflow that uses `flights_rec1` and the model you specified.

```r
wflow1 <- workflow() %>%
  add_model(spec) %>%
  add_recipe(flights_rec1)
```

6. Fit the this model to the training data using your workflow and display a tidy summary of the model fit.

```r
flightsFit1 <- wflow1 %>%
  fit(train)

tidy(flightsFit1) %>%
  kable(digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	2.170	3.794	0.572	0.567
dep_time	-0.002	0.000	-26.043	0.000
air_time	-0.033	0.002	-16.029	0.000
distance	0.004	0.000	15.764	0.000
date	0.000	0.000	0.572	0.567
origin_JFK	0.148	0.089	1.665	0.096
origin_LGA	0.017	0.080	0.214	0.830
carrier_AA	0.278	0.149	1.871	0.061
carrier_AS	0.965	0.777	1.242	0.214
carrier_B6	-0.134	0.123	-1.094	0.274
carrier_DL	0.343	0.136	2.517	0.012
carrier_EV	-0.281	0.138	-2.033	0.042
carrier_F9	-0.353	0.498	-0.709	0.478
carrier_FL	-0.263	0.283	-0.928	0.354
carrier_HA	-1.070	1.088	-0.983	0.326
carrier_MQ	0.002	0.143	0.011	0.991
carrier_OO	-11.499	119.468	-0.096	0.923
carrier_UA	0.129	0.147	0.874	0.382
carrier_US	0.607	0.179	3.394	0.001

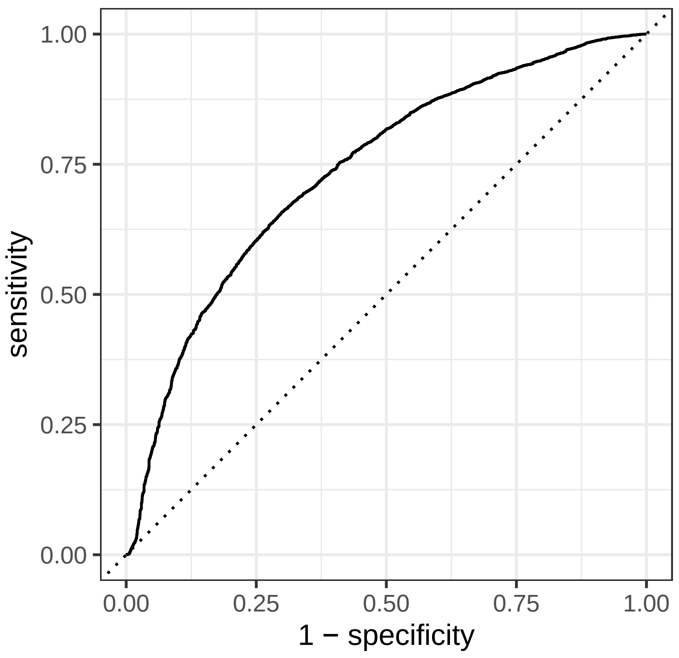
term	estimate	std.error	statistic	p.value
carrier_VX	-0.198	0.250	-0.794	0.427
carrier_WN	0.133	0.185	0.718	0.473
carrier_YV	0.288	0.509	0.567	0.571

7. Predict `arr_delay` for the testing data using this model.

```
flightsPred1 <- predict(flightsFit1, test, type = "prob") %>%
  bind_cols(test)
```

8. Plot the ROC curve and find the area under the curve. Comment on how well you think this model has done for predicting arrival delay.

```
flightsPred1 %>%
  roc_curve(truth = arr_delay, .pred_on_time, event_level = "second") %>%
  autoplot()
```



```
flightsPred1 %>%
  roc_auc(truth = arr_delay, .pred_on_time, event_level = "second")
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 roc_auc binary         0.737
```

This model does okay at predicting arrival delay. With an AUC of 0.737, it's definitely capturing a lot of the delayed flights, but it has a fairly high false positive rate.

Model 2: Let's be a bit more thoughtful (10 minutes)

9. Define a new recipe, `flights_rec2`, that, in addition to what was done in `flights_rec1`, adds features for day of week and month based on `date` and also adds indicators for all US holidays (also based on `date`). A list of these holidays can be found in `timeDate::listHolidays("US")`. Once these features are added, `date` should be removed from the data. Then, create a new workflow, fit the same model (logistic regression) to the training data, and do predictions on the testing data. Finally, draw another ROC curve and find the area under the curve.

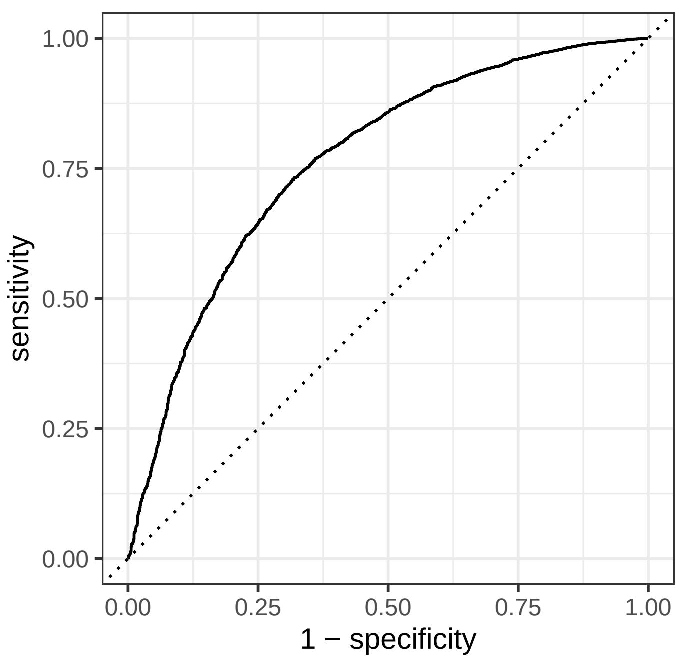
```
flights_rec2 <- recipe(arr_delay ~ dep_time + origin + air_time + distance + carrier + dat
  step_date(date, features = c("dow", "month")) %>%
  step_holiday(date, holidays = c(timeDate::listHolidays("US"))) %>%
  step_rm(date) %>%
  #step_other(all_nominal_predictors()) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors())

wflow2 <- workflow() %>%
  add_model(spec) %>%
  add_recipe(flights_rec2)

flightsFit2 <- wflow2 %>%
  fit(train)

flightsPred2 <- predict(flightsFit2, test, type = "prob") %>%
  bind_cols(test)

flightsPred2 %>%
  roc_curve(truth = arr_delay, .pred_on_time, event_level = "second") %>%
  autoplot()
```



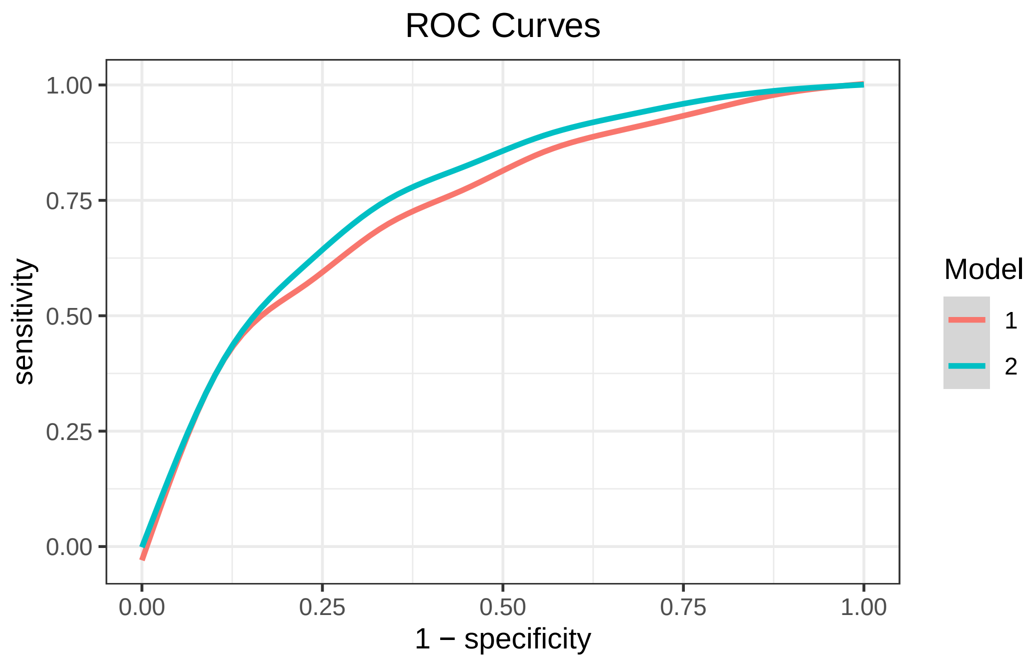
```r
flightsPred2 %>%
  roc_auc(truth = arr_delay, .pred_on_time, event_level = "second")
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 roc_auc binary         0.765
```

Putting it altogether (10 minutes)

10. Create an ROC curve that plots both models, in different colors, and adds a legend indicating which model is which.

```r
full <- flightsPred1 %>%
  roc_curve(truth = arr_delay, .pred_on_time, event_level = "second") %>%
  mutate(curve = 1)
full2 <- flightsPred2 %>%
  roc_curve(truth = arr_delay, .pred_on_time, event_level = "second") %>%
  mutate(curve = 2)
full <- full %>%
  bind_rows(full2) %>%
  mutate(curve = factor(curve))
ggplot(full, aes(x = 1 - specificity, y = sensitivity, color = curve)) +
  geom_smooth() +
  theme_bw() +
  labs(title = "ROC Curves", color = "Model") +
  theme(plot.title = element_text(hjust = 0.5))
```



11. Compare the predictive performance of this new model to the previous one. Based on the ROC curves and area under the curve statistic, which model does better?

Model 2 is slightly better than model 1. It has a larger AUC and the sensitivity increases faster relative to the false positive rate.

! 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-14-` repo on GitHub. (You do not submit AEs on Gradescope).

Acknowledgement

This exercise was inspired by tidymodels.org/start/recipes and adapted from sta210-s22.github.io/website/ae/ae-10-flight-delays.