AE 10: Model comparison

Add your name here

Packages

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

Data

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

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

```
Rows: 25000 Columns: 10
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (4): origin, dest, carrier, arr_delay
dbl  (4): dep_time, flight, air_time, distance
dttm (1): time_hour
date (1): date

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.
```

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.

```r
flight_data <- flight_data %>%
  mutate(arr_delay = as.factor(arr_delay))
```

2. Let's get started with some data prep: Convert all variables that are character strings to factors.

```r
flight_data <- flight_data %>%
  mutate(across(where(is.character), as.factor))
```

Modeling prep

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

```r
set.seed(222)

flight_data <- initial_split(flight_data, prop = 3/4)

flight_train <- training(flight_data)
flight_test  <- testing(flight_data)
```

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

```r
flight_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

5. 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. 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 ~ ., data = flight_train) %>%
  update_role(flight, time_hour, new_role = "ID") %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors())
```

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

```
flights_wflow1 <- workflow() %>%
  add_model(flight_spec) %>%
  add_recipe(flights_rec1)

flights_wflow1
```

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

-- Preprocessor ----------------------------------------------------------------
2 Recipe Steps

* step_dummy()
* step_zv()

-- Model -----------------------------------------------------------------------
Logistic Regression Model Specification (classification)

Computational engine: glm
```

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

```
flight_fit1 <- flights_wflow1 %>%
  fit(data = flight_train)

flight_fit1 %>%
  tidy()
```

```
# A tibble: 119 x 5
   term         estimate   std.error statistic   p.value
   <chr>           <dbl>       <dbl>     <dbl>     <dbl>
 1 (Intercept)  13.3      287.           0.0464  9.63e-  1
 2 dep_time     -0.00164    0.0000504 -32.6      1.04e-233
 3 air_time     -0.0349     0.00179   -19.5      1.75e- 84
 4 distance      0.00533    0.00523     1.02     3.08e-  1
 5 date          0.000227   0.000198    1.15     2.51e-  1
 6 origin_JFK    0.0830     0.102       0.815    4.15e-  1
```

```
 7 origin_LGA   -0.0360     0.0983     -0.366   7.14e-  1
 8 dest_ACK    -12.4      287.         -0.0434  9.65e-  1
 9 dest_ALB    -12.4      287.         -0.0433  9.65e-  1
10 dest_ANC     -3.75     928.         -0.00404 9.97e-  1
# ... with 109 more rows
```

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

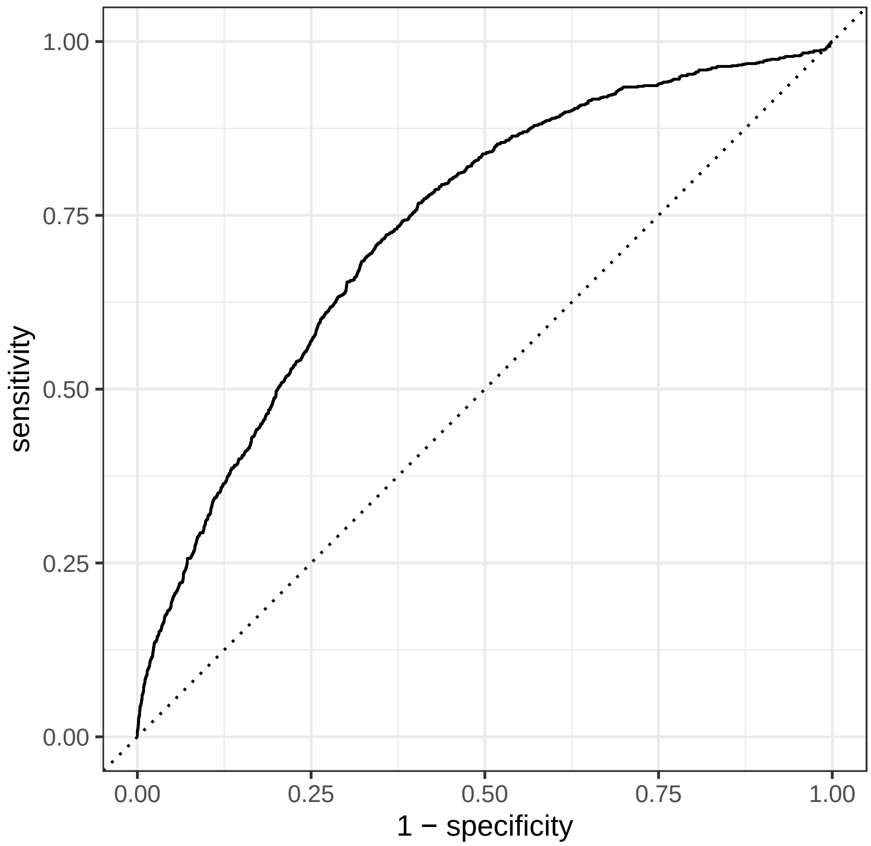
```r
flights_aug1 <- augment(flight_fit1, flight_test)

flights_aug1 %>%
  select(arr_delay, time_hour, flight, .pred_class, .pred_on_time)
```

```
# A tibble: 6,250 x 5
   arr_delay time_hour           flight .pred_class .pred_on_time
   <fct>     <dttm>               <dbl> <fct>               <dbl>
 1 on_time   2013-09-12 20:00:00   1178 on_time             0.843
 2 on_time   2013-11-23 22:00:00    359 on_time             0.847
 3 late      2013-03-17 22:00:00    124 on_time             0.630
 4 on_time   2013-12-05 20:00:00   1638 on_time             0.898
 5 late      2013-04-02 00:00:00   5681 late                0.450
 6 on_time   2013-11-05 23:00:00   2915 on_time             0.666
 7 on_time   2013-05-21 10:00:00    413 on_time             0.968
 8 on_time   2013-04-05 22:00:00   4277 on_time             0.789
 9 late      2013-03-13 01:00:00    515 on_time             0.965
10 on_time   2013-03-27 10:00:00    303 on_time             0.931
# ... with 6,240 more rows
```

9. 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.

```r
flights_aug1 %>%
  roc_curve(truth = arr_delay, .pred_late) %>%
  autoplot()
```




```
flights_aug1 %>%
  roc_auc(truth = arr_delay, .pred_late)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 roc_auc binary         0.734
```

Model 2: Let's be a bit more thoughtful

10. 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. Compare the predictive performance of

this new model to the previous one. Based on the area under the curve statistic, which model does better?

```r
flights_rec2 <- recipe(arr_delay ~ ., data = flight_train) %>%
  update_role(flight, time_hour, new_role = "ID") %>%
  step_date(date, features = c("dow", "month")) %>%
  step_holiday(date,
               holidays = timeDate::listHolidays("US"),
               keep_original_cols = FALSE) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors())

flights_wflow2 <- workflow() %>%
  add_model(flight_spec) %>%
  add_recipe(flights_rec2)

flights_fit2 <- flights_wflow2 %>%
  fit(data = flight_train)

flights_fit2 %>%
  tidy()
```

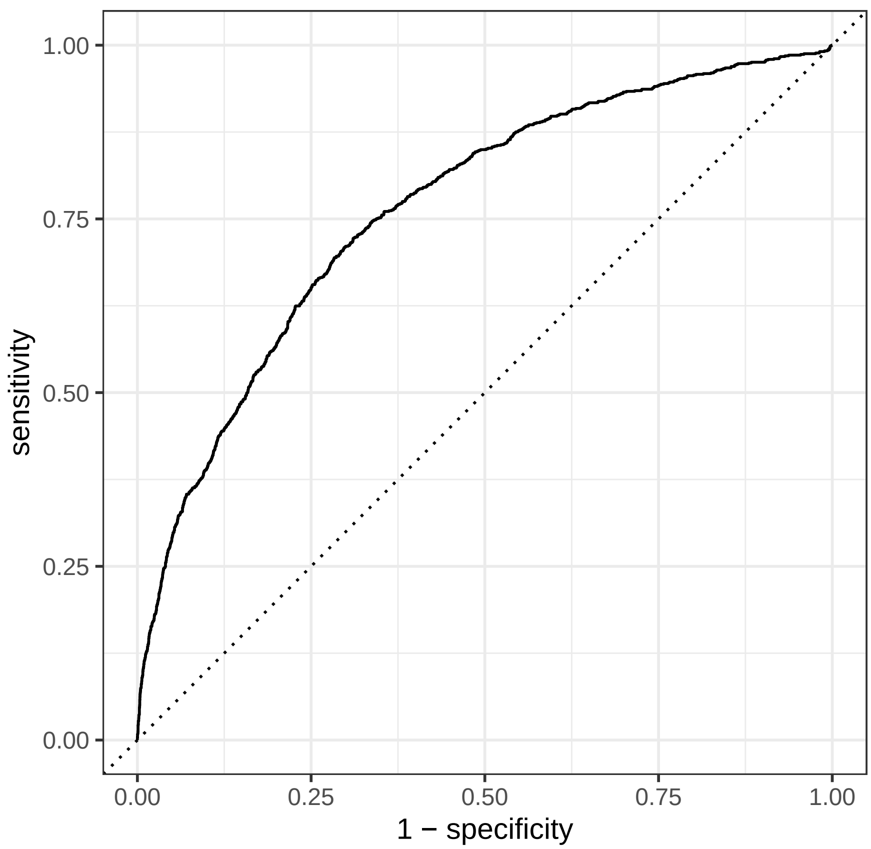
```
# A tibble: 152 x 5
   term                        estimate   std.error statistic   p.value
   <chr>                          <dbl>       <dbl>     <dbl>     <dbl>
 1 (Intercept)                 19.0     282.           0.0674 9.46e-  1
 2 dep_time                    -0.00168   0.0000512  -32.8    1.52e-236
 3 air_time                    -0.0447    0.00205    -21.8    1.60e-105
 4 distance                     0.00596   0.00535      1.11   2.65e-  1
 5 date_USChristmasDay          0.955     0.634        1.51   1.32e-  1
 6 date_USColumbusDay           0.488     0.621        0.787  4.31e-  1
 7 date_USCPulaskisBirthday     0.187     0.411        0.455  6.49e-  1
 8 date_USDecorationMemorialDay 0.380     0.363        1.05   2.95e-  1
 9 date_USElectionDay           0.643     0.617        1.04   2.97e-  1
10 date_USGoodFriday            0.762     0.489        1.56   1.19e-  1
# ... with 142 more rows
```

```r
flights_aug2 <- augment(flights_fit2, flight_test)

flights_aug2 %>%
  select(arr_delay, time_hour, flight, .pred_class, .pred_on_time)
```

```
# A tibble: 6,250 x 5
   arr_delay time_hour           flight .pred_class .pred_on_time
   <fct>     <dttm>               <dbl> <fct>               <dbl>
 1 on_time   2013-09-12 20:00:00   1178 on_time             0.884
 2 on_time   2013-11-23 22:00:00    359 on_time             0.938
 3 late      2013-03-17 22:00:00    124 on_time             0.690
 4 on_time   2013-12-05 20:00:00   1638 on_time             0.868
 5 late      2013-04-02 00:00:00   5681 late                0.392
 6 on_time   2013-11-05 23:00:00   2915 on_time             0.864
 7 on_time   2013-05-21 10:00:00    413 on_time             0.965
 8 on_time   2013-04-05 22:00:00   4277 on_time             0.753
 9 late      2013-03-13 01:00:00    515 on_time             0.965
10 on_time   2013-03-27 10:00:00    303 on_time             0.940
# ... with 6,240 more rows
```

```r
flights_aug2 %>%
  roc_curve(truth = arr_delay, .pred_late) %>%
  autoplot()
```



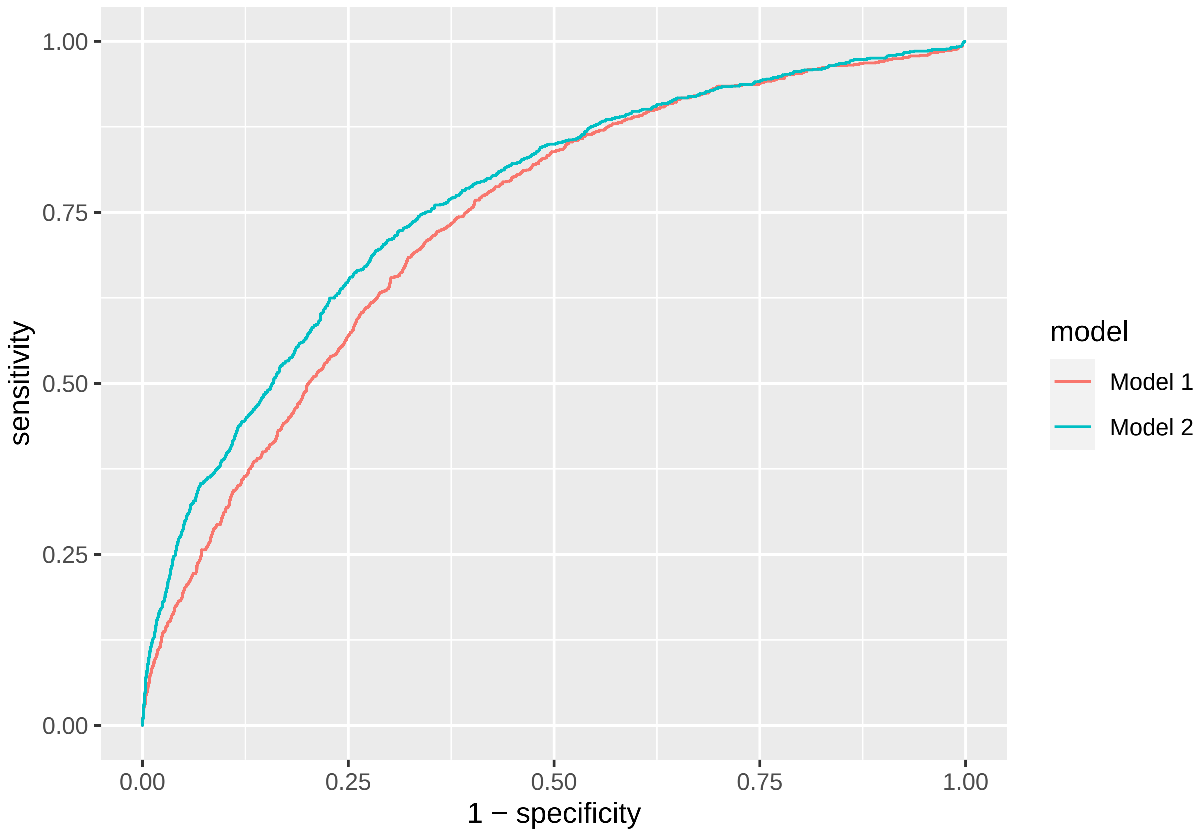
```
flights_aug2 %>%
  roc_auc(truth = arr_delay, .pred_late)
```

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

Putting it altogether

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

```
flights_aug1 %>% roc_curve(truth = arr_delay, .pred_late) %>% mutate(model = "Model 1") %>%
  bind_rows(flights_aug2 %>% roc_curve(truth = arr_delay, .pred_late) %>% mutate(model = "Mod
  ggplot(aes(x = 1 - specificity, y = sensitivity, color = model)) +
  geom_line()
```



Acknowledgement

This exercise was inspired by https://www.tidymodels.org/start/recipes/.