Tune better models to predict children in hotel bookings

Suggested answers

APPLICATION EXERCISE ANSWERS

MODIFIED
November 7, 2024

Your Turn 1

Fill in the blanks to return the accuracy and ROC AUC for this model using 10-fold cross-validation.

```
tree_mod <- decision_tree(engine = "rpart") |>
  set_mode("classification")

tree_wf <- workflow() |>
  add_formula(children ~ .) |>
  add_model(tree_mod)
```

Fill in the blanks to return the accuracy and ROC AUC for this model using 10-fold cross-validation.

```
set.seed(100)
______ |>
  ______(resamples = hotels_folds) |>
  ______
```

Answer:

```
set.seed(100)
tree_wf |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

```
# A tibble: 3 × 6
  .metric     .estimator  mean     n std_err .config
  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy    binary     0.773    10 0.00567 Preprocessor1_Model1
2 brier_class binary     0.158    10 0.00322 Preprocessor1_Model1
3 roc_auc     binary     0.832    10 0.00672 Preprocessor1_Model1
```

Your Turn 2

Create a new parsnip model called `rf_mod`, which will learn an ensemble of classification trees from our training data using the **ranger** package. Update your `tree_wf` with this new model.

Fit your workflow with 10-fold cross-validation and compare the ROC AUC of the random forest to your single decision tree model — which predicts the test set better?

Hint: you'll need https://www.tidymodels.org/find/parsnip/

```
# model
rf_mod <- _____ |>
  _____("ranger") |>
  _____("classification")

# workflow
rf_wf <- tree_wf |>
  update_model(_____)

# fit with cross-validation
set.seed(100)
_____ |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

Answer:

```
# model
rf_mod <- rand_forest(engine = "ranger") |>
  set_mode("classification")

# workflow
rf_wf <- tree_wf |>
  update_model(rf_mod)

# fit with cross-validation
set.seed(100)
rf_wf |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

```
# A tibble: 3 × 6
  .metric     .estimator  mean     n std_err .config
  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy    binary     0.829    10 0.00332 Preprocessor1_Model1
2 brier_class binary     0.123    10 0.00176 Preprocessor1_Model1
3 roc_auc     binary     0.912    10 0.00320 Preprocessor1_Model1
```

Your Turn 3

Challenge: Fit 3 more random forest models, each using 5, 12, and 21 variables at each split. Update your `rf_wf` with each new model. Which value maximizes the area under the ROC curve?

```
rf5_mod <- rf_mod |>
  set_args(mtry = 5)

rf12_mod <- rf_mod |>
  set_args(mtry = 12)

rf21_mod <- rf_mod |>
  set_args(mtry = 21)
```

Do this for each model above:

```
_____ <- rf_wf |>
  update_model(_____)

set.seed(100)
_____ |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

Answer:

```
# 5
rf5_wf <- rf_wf |>
  update_model(rf5_mod)

set.seed(100)
rf5_wf |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

```
# A tibble: 3 × 6
  .metric     .estimator  mean     n std_err .config
  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy    binary     0.829    10 0.00376 Preprocessor1_Model1
2 brier_class binary     0.122    10 0.00176 Preprocessor1_Model1
3 roc_auc     binary     0.912    10 0.00305 Preprocessor1_Model1
```

```
# 12
rf12_wf <- rf_wf |>
  update_model(rf12_mod)

set.seed(100)
rf12_wf |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

```
# A tibble: 3 × 6
  .metric     .estimator  mean     n std_err .config
  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy    binary     0.831    10 0.00414 Preprocessor1_Model1
2 brier_class binary     0.123    10 0.00239 Preprocessor1_Model1
3 roc_auc     binary     0.908    10 0.00418 Preprocessor1_Model1
```

```
# 21
rf21_wf <- rf_wf |>
  update_model(rf21_mod)

set.seed(100)
rf21_wf |>
  fit_resamples(resamples = hotels_folds) |>
  collect_metrics()
```

```
# A tibble: 3 × 6
  .metric     .estimator  mean     n std_err .config
  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy    binary     0.827    10 0.00382 Preprocessor1_Model1
2 brier_class binary     0.125    10 0.00256 Preprocessor1_Model1
3 roc_auc     binary     0.905    10 0.00438 Preprocessor1_Model1
```

Your Turn 4

Edit the random forest model to tune the `mtry` and `min_n` hyper-parameters; call the new model spec `rf_tuner`.

Update your workflow to use the tuned model.

Then use `tune_grid()` to find the best combination of hyper-parameters to maximize `roc_auc`; let tune set up the grid for you.

How does it compare to the average ROC AUC across folds from `fit_resamples()`?

```
rf_mod <- rand_forest(engine = "ranger") |>
  set_mode("classification")

rf_wf <- workflow() |>
  add_formula(children ~ .) |>
  add_model(rf_mod)

set.seed(100) # Important!
rf_results <- rf_wf |>
  fit_resamples(resamples = hotels_folds,
                metrics = metric_set(roc_auc),
                # change me to control_grid() with tune_grid
                control = control_resamples(save_workflow = TRUE))
```

```r
rf_results |>
  collect_metrics()
```

```
# A tibble: 1 × 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 roc_auc binary     0.912    10 0.00320 Preprocessor1_Model1
```

Answer:

```r
rf_tuner <- rand_forest(
    engine = "ranger",
    mtry = tune(),
    min_n = tune()
  ) |>
  set_mode("classification")

rf_wf <- rf_wf |>
  update_model(rf_tuner)

set.seed(100) # Important!
rf_results <- rf_wf |>
  tune_grid(resamples = hotels_folds,
            control = control_grid(save_workflow = TRUE))
```

```
i Creating pre-processing data to finalize unknown parameter: mtry
```

Your Turn 5

Use `fit_best()` to take the best combination of hyper-parameters from `rf_results` and use them to predict the test set.

How does our actual test ROC AUC compare to our cross-validated estimate?

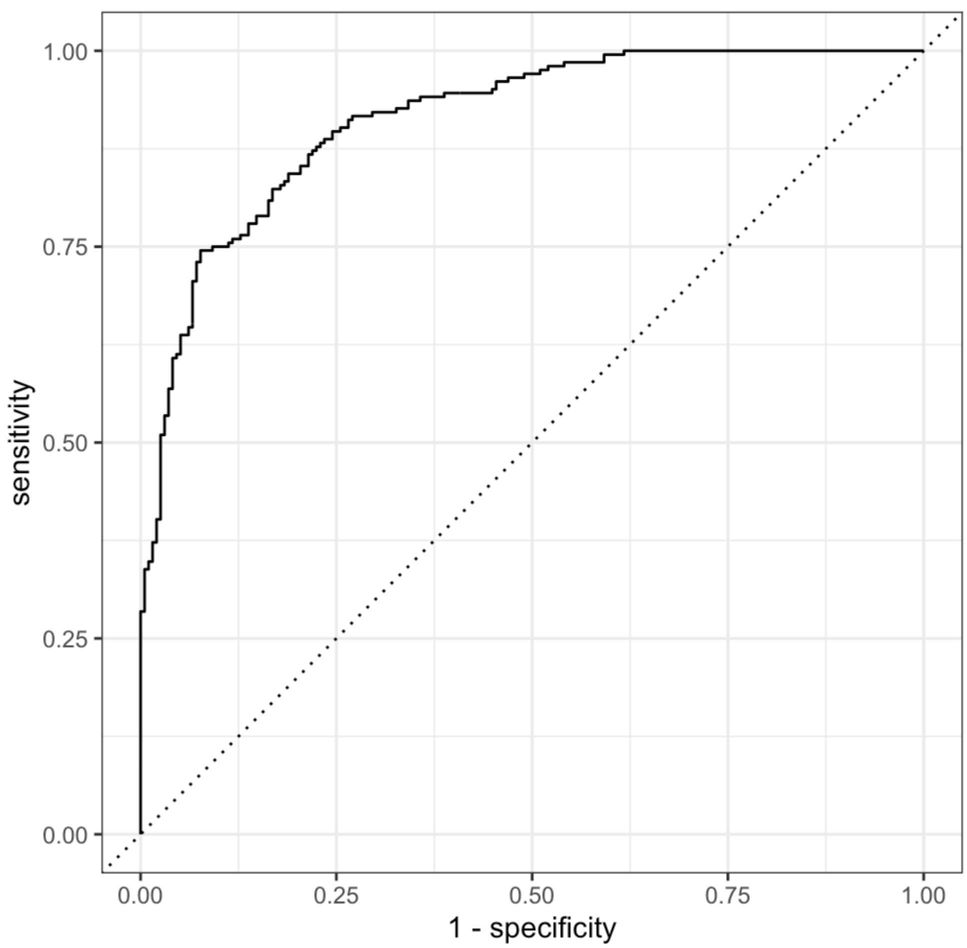
```r
hotels_best <- fit_best(rf_results)

# cross validated ROC AUC
rf_results |>
  show_best(metric = "roc_auc", n = 5)
```

```
# A tibble: 5 × 8
   mtry min_n .metric .estimator  mean     n std_err .config
  <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1     3    15 roc_auc binary     0.910    10 0.00283 Preprocessor1_Model07
2     8    20 roc_auc binary     0.909    10 0.00376 Preprocessor1_Model10
3     7    36 roc_auc binary     0.908    10 0.00372 Preprocessor1_Model02
4     9    28 roc_auc binary     0.907    10 0.00381 Preprocessor1_Model01
5    12    21 roc_auc binary     0.907    10 0.00430 Preprocessor1_Model03
```

```
# test set ROC AUC
augment(hotels_best, new_data = hotels_test) |>
  roc_auc(truth = children, .pred_children)
```

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 roc_auc binary         0.913
```

```
# test set ROC curve
augment(hotels_best, new_data = hotels_test) |>
  roc_curve(truth = children, .pred_children) |>
  autoplot()
```



Acknowledgments

- Materials derived from Tidymodels, Virtually: An Introduction to Machine Learning with Tidymodels by Allison Hill.
- Dataset and some modeling steps derived from A predictive modeling case study and licensed under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA) License.

Session information