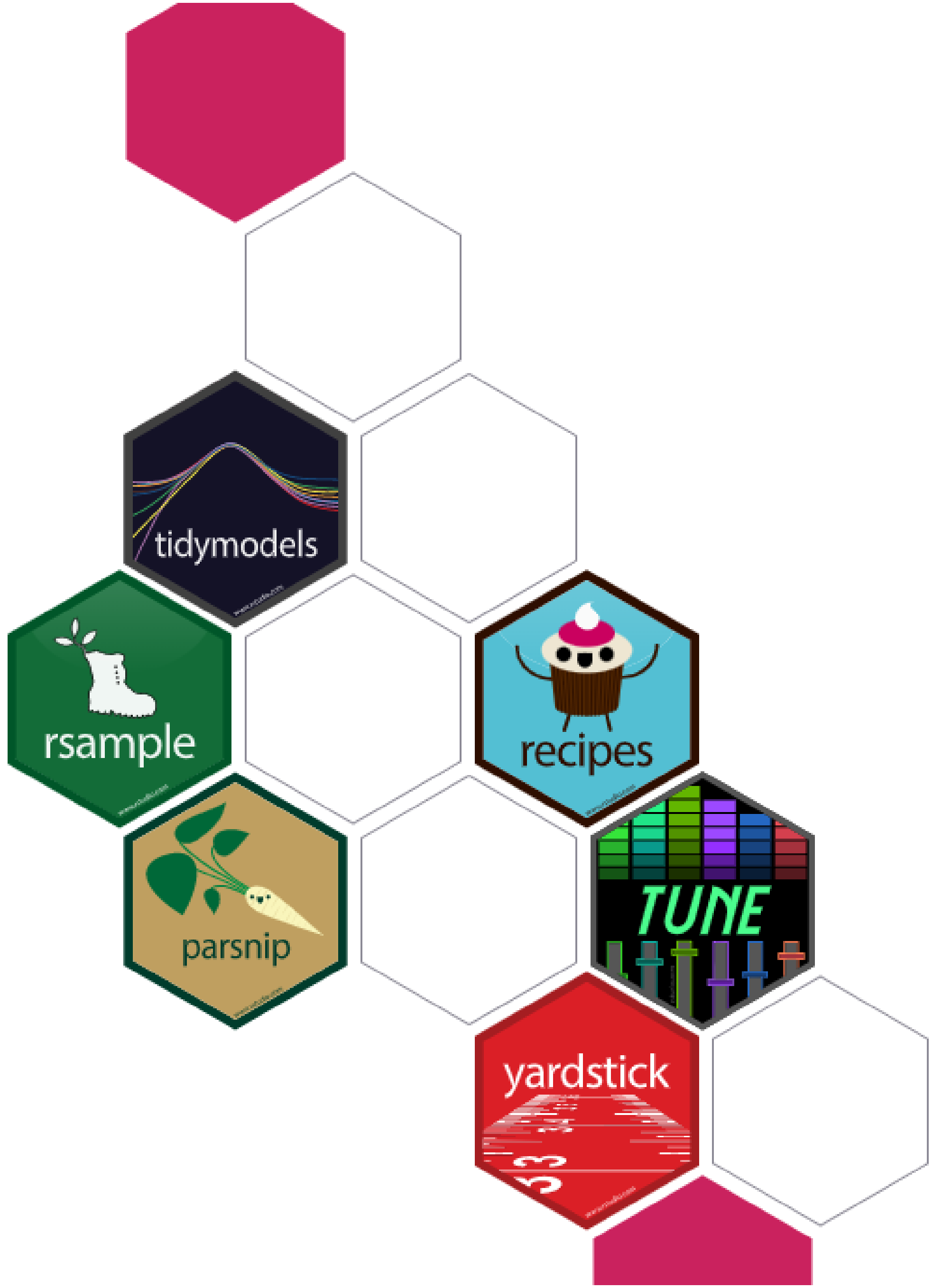
Tidymodels

Lecture 22

Dr. Colin Rundel


tidymodels
rsample
recipes
parsnip
TUNE
yardstick

Tidymodels

```r
library(tidymodels)
```

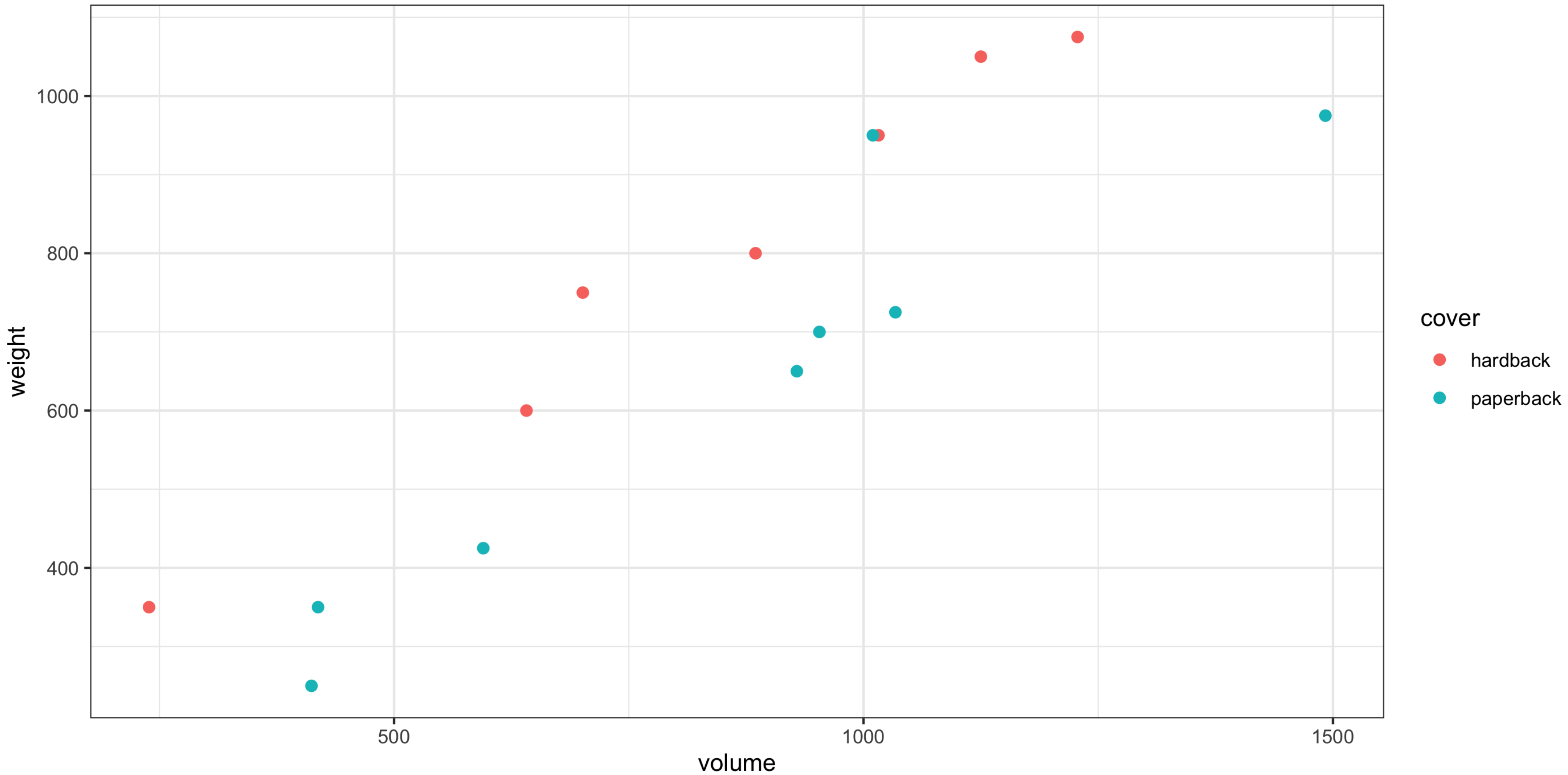
```
── Attaching packages ──────────────────────────────────────────── tidymodels
1.1.1 ──

✔ broom        1.0.5     ✔ rsample      1.2.0
✔ dials        1.2.0     ✔ tune         1.1.2
✔ infer        1.0.5     ✔ workflows    1.1.3
✔ modeldata    1.2.0     ✔ workflowsets 1.0.1
✔ parsnip      1.1.1     ✔ yardstick    1.2.0
✔ recipes      1.0.8

── Conflicts ─────────────────────────────────────────────
tidymodels_conflicts() ──
✖ scales::discard()   masks purrr::discard()
✖ dplyr::filter()     masks stats::filter()
✖ recipes::fixed()    masks stringr::fixed()
✖ dplyr::lag()        masks stats::lag()
✖ rsample::populate() masks Rcpp::populate()
✖ yardstick::spec()   masks readr::spec()
```

Book data

```r
(books = DAAG::allbacks |>
  as_tibble() |>
  select(-area) |>
  mutate(
    cover = forcats::fct_recode(
      cover,
      "hardback" = "hb",
      "paperback" = "pb"
    )
  )
)
```

```
# A tibble: 15 × 3
   volume weight cover
    <dbl>  <dbl> <fct>
 1    885    800 hardback
 2   1016    950 hardback
 3   1125   1050 hardback
 4    239    350 hardback
 5    701    750 hardback
 6    641    600 hardback
 7   1228   1075 hardback
 8    412    250 paperback
 9    953    700 paperback
10    929    650 paperback
```

```r
ggplot(books, aes(x=volume, y=weight, color = cover)) +
  geom_point(size=2)
```



Building a tidymodel

```r
linear_reg()
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Building a tidymodel

```
linear_reg() |>
  set_engine("lm")
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Building a tidymodel

```r
linear_reg() |>
  set_engine("lm") |>
  fit(weight ~ volume * cover, data = books)
```

```
parsnip model object


Call:
stats::lm(formula = weight ~ volume * cover, data = 
data)

Coefficients:
          (Intercept)                 volume  
coverpaperback  
            161.58654                0.76159  
-120.21407  
volume:coverpaperback  
             -0.07573  
```

```r
lm(weight ~ volume * cover, data = books)
```

```
Call:
lm(formula = weight ~ volume * cover, data = books)

Coefficients:
          (Intercept)                 volume  
coverpaperback  
            161.58654                0.76159  
-120.21407  
volume:coverpaperback  
             -0.07573  
```

broom

www.tidyverse.org

Tidy model objects

```r
lm_b = lm(weight ~ volume * cover, data = books)
  summary()
```

```r
lm_tm = linear_reg() |>
  set_engine("lm") |>
  fit(weight ~ volume * cover, data = books)
```

```r
summary(lm_b)
```

```
              Length Class  Mode
call           3     -none- call
terms          3     terms  call
residuals     15     -none- numeric
coefficients  16     -none- numeric
aliased        4     -none- logical
sigma          1     -none- numeric
df             3     -none- numeric
r.squared      1     -none- numeric
adj.r.squared  1     -none- numeric
fstatistic     3     -none- numeric
cov.unscaled  16     -none- numeric
```

```r
summary(lm_tm)
```

```
             Length Class      Mode
lvl           0     -none-     NULL
spec          7     linear_reg list
fit          13     lm         list
preproc       1     -none-     list
elapsed       1     -none-     list
censor_probs  0     -none-     list
```

```r
broom::tidy(lm_tm)
```

```
# A tibble: 4 × 5
  term                  estimate std.error statistic    p.value
  <chr>                    <dbl>     <dbl>     <dbl>      <dbl>
1 (Intercept)           162.       86.5        1.87  0.0887
2 volume                  0.762     0.0972     7.84  0.00000794
3 coverpaperback       -120.      116.        -1.04  0.321
4 volume:coverpaperback  -0.0757    0.128     -0.592 0.566
```

Tidy model statistics

```
broom::glance(lm_b)
```

```
# A tibble: 1 × 8
  r.squared adj.r.squared sigma statistic    p.value    df df.residual  nobs
      <dbl>         <dbl> <dbl>     <dbl>      <dbl> <dbl>       <int> <dbl>
1     0.930         0.911  80.4      48.5 0.00000124     3          11    15
```

```
broom::glance(lm_tm)
```

```
# A tibble: 1 × 12
  r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC deviance
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>    <dbl>
1     0.930         0.911  80.4      48.5   1.24e-6     3  -84.8  180.  183.   71118.
# ℹ 2 more variables: df.residual <int>, nobs <int>
```

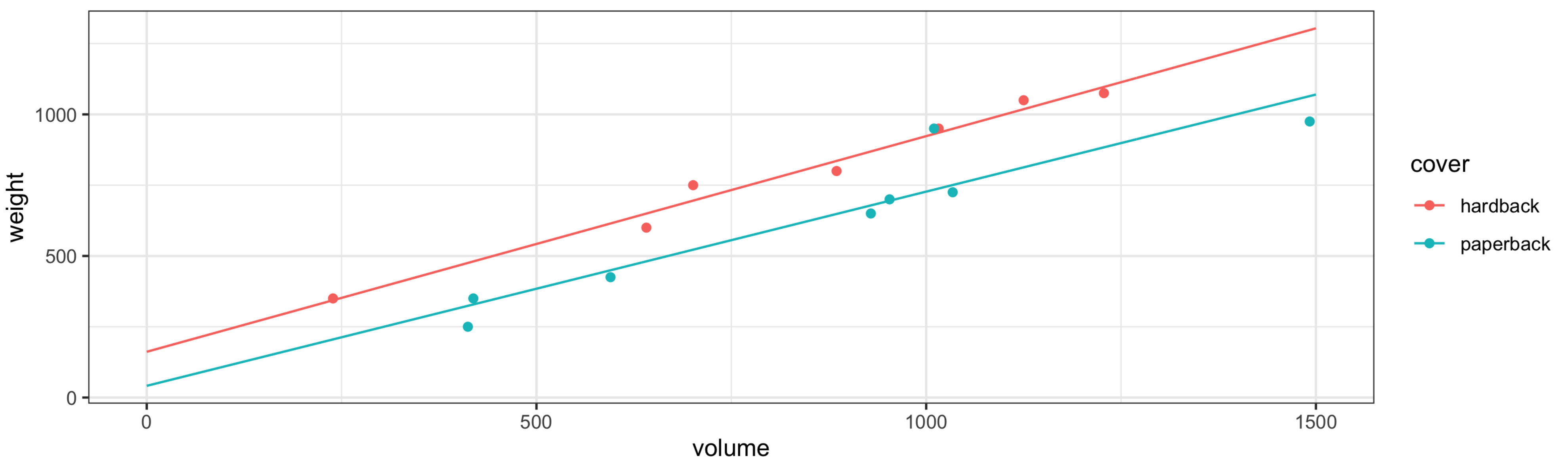
Tidy model prediction

```r
broom::augment(lm_tm, new_data = books)
```

```
# A tibble: 15 × 5
   .pred .resid volume weight cover
   <dbl>  <dbl>  <dbl>  <dbl> <fct>
 1  836. -35.6     885    800 hardback
 2  935.  14.6    1016    950 hardback
 3 1018.  31.6    1125   1050 hardback
 4  344.   6.39    239    350 hardback
 5  695.  54.5     701    750 hardback
 6  650. -49.8     641    600 hardback
 7 1097. -21.8    1228   1075 hardback
 8  324. -73.9     412    250 paperback
 9  695.   5.00    953    700 paperback
```

Putting it together

```
lm_tm |>
  augment(
    new_data = tidyr::expand_grid(
      volume = seq(0, 1500, by=5),
      cover = c("hardback", "paperback") |> as.factor()
    )
  ) |>
  rename(weight = .pred) |>
  ggplot(aes(x = volume, y = weight, color = cover, group = cover)) +
    geom_line() +
    geom_point(data = books)
```



parsnip

www.rstudio.com

Why do we care?

```r
show_engines("linear_reg")
```

```
# A tibble: 7 × 2
  engine mode
  <chr>  <chr>
1 lm     regression
2 glm    regression
3 glmnet regression
4 stan   regression
5 spark  regression
6 keras  regression
7 brulee regression
```

```r
(bayes_tm = linear_reg() |>
  set_engine(
    "stan",
    prior_intercept = rstanarm::student_t(df = 1
    prior = rstanarm::student_t(df = 1),
    seed = 1234
  )
)
```

```
Linear Regression Model Specification (regression)

Engine-Specific Arguments:
  prior_intercept = rstanarm::student_t(df = 1)
  prior = rstanarm::student_t(df = 1)
  seed = 1234

Computational engine: stan
```

Fitting with `rstanarm`

```
(bayes_tm = bayes_tm |>
  fit(weight ~ volume * cover, data = books)
)
```

```
parsnip model object

stan_glm
 family:       gaussian [identity]
 formula:      weight ~ volume * cover
 observations: 15
 predictors:   4
------
                     Median MAD_SD
(Intercept)          95.6   61.8
volume                0.8    0.1
coverpaperback       -0.2    3.7
volume:coverpaperback -0.2    0.1

Auxiliary parameter(s):
      Median MAD_SD
sigma 85.5   17.8
```

See `?details_linear_reg_stan` for details within `parsnip`

What was actually run?

```r
linear_reg() |>
  set_engine(
    "stan",
    prior_intercept = rstanarm::student_t(df = 1),
    prior = rstanarm::student_t(df = 1),
    seed = 1234
  ) |>
  translate()
```

```
Linear Regression Model Specification (regression)

Engine-Specific Arguments:
  prior_intercept = rstanarm::student_t(df = 1)
  prior = rstanarm::student_t(df = 1)
  seed = 1234

Computational engine: stan

Model fit template:
rstanarm::stan_glm(formula = missing_arg(), data = missing_arg(),
    weights = missing_arg(), prior_intercept = rstanarm::student_t(df = 1),
    prior = rstanarm::student_t(df = 1), seed = 1234, family = stats::gaussian,
    refresh = 0)
```

Back to broom

```
broom::tidy(bayes_tm)
```

```
Error in warn_on_stanreg(x): The supplied model object seems to be outputted from the rstanarm package.
Tidiers for mixed model output now live in the broom.mixed package.
```

```
broom.mixed::tidy(bayes_tm)
```

```
# A tibble: 4 × 3
  term                  estimate std.error
  <chr>                    <dbl>     <dbl>
1 (Intercept)             95.6      61.8
2 volume                   0.831     0.0743
3 coverpaperback          -0.238     3.73
4 volume:coverpaperback   -0.198     0.0506
```

```
broom.mixed::glance(bayes_tm)
```

```
# A tibble: 1 × 4
  algorithm   pss  nobs sigma
  <chr>     <dbl> <int> <dbl>
1 sampling   4000    15  85.5
```

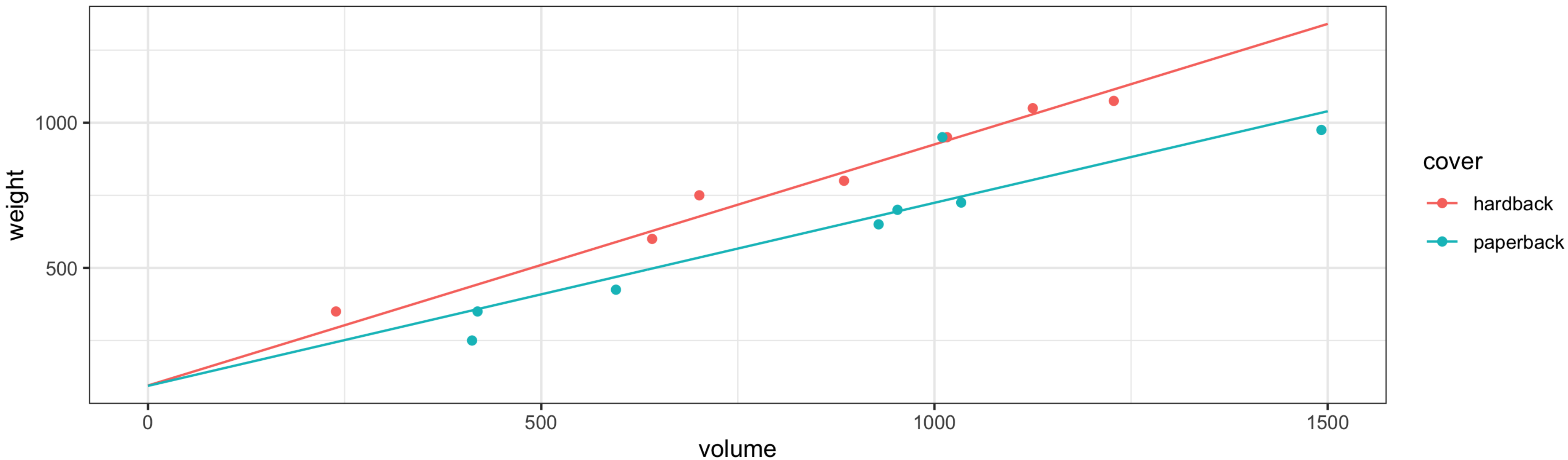
Augment

```
augment(bayes_tm, new_data=books)
```

```
# A tibble: 15 × 5
    .pred  .resid volume weight cover
    <dbl>   <dbl>  <dbl>  <dbl> <fct>
 1  830.   -29.8     885    800 hardback
 2  939.    11.5    1016    950 hardback
 3 1029.    21.0    1125   1050 hardback
 4  294.    56.4     239    350 hardback
 5  677.    73.0     701    750 hardback
 6  627.   -27.2     641    600 hardback
 7 1114.   -39.5    1228   1075 hardback
 8  354.  -104.      412    250 paperback
 9  695.     5.39    953    700 paperback
```

Predictions

```r
bayes_tm |>
  augment(
    new_data = tidyr::expand_grid(
      volume = seq(0, 1500, by=5),
      cover = c("hardback", "paperback") |> as.factor()
    )
  ) |>
  rename(weight = .pred) |>
  ggplot(aes(x = volume, y = weight, color = cover, group = cover)) +
    geom_line() +
    geom_point(data = books)
```



Performance

```r
lm_tm |>
  augment(new_data = books) |>
  yardstick::rmse(weight, .pred)
```

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard        68.9
```

```r
bayes_tm |>
  augment(new_data = books) |>
  yardstick::rmse(weight, .pred)
```

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard        72.1
```

More on combining these type of results later, see `workflows`

Cross validation and Feature engineering

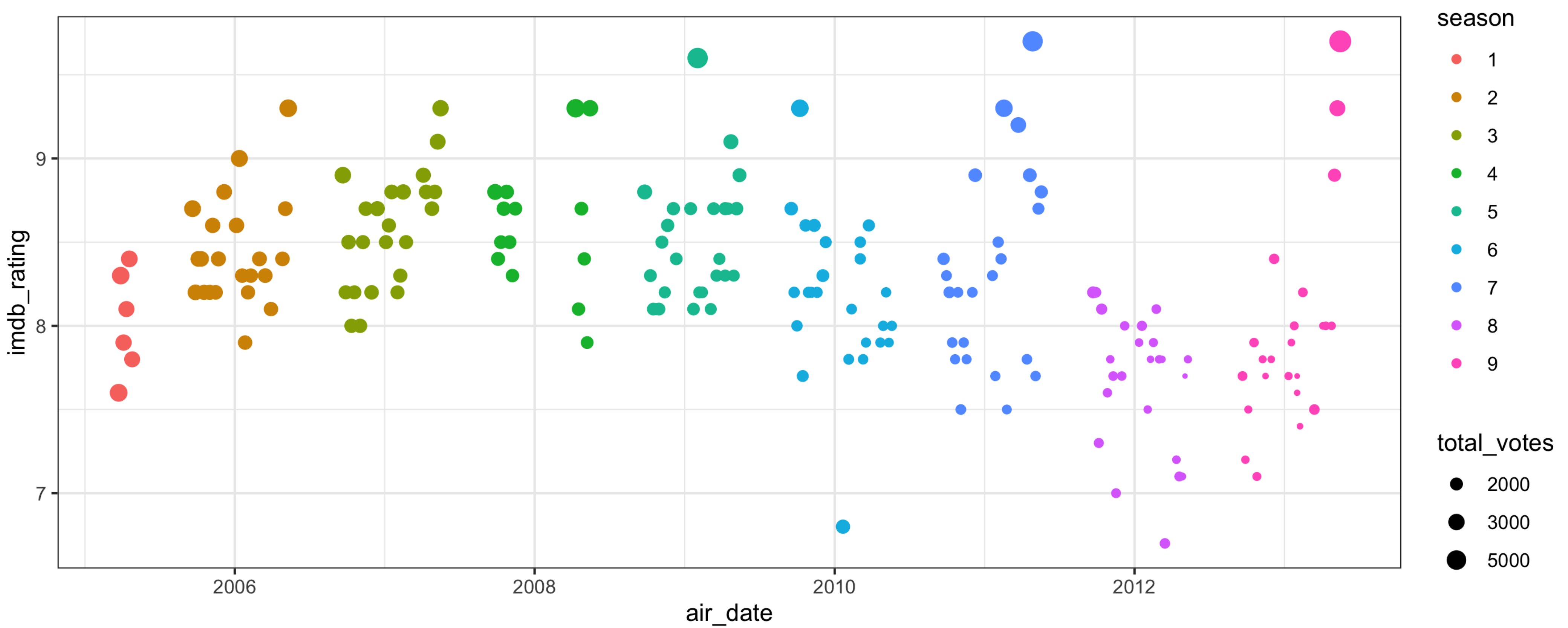
The Office & IMDB

```r
(office_ratings = read_csv("data/office_ratings.csv"))
```

```
# A tibble: 188 × 6
   season episode title                 imdb_rating total_votes air_date
    <dbl>   <dbl> <chr>                       <dbl>       <dbl> <date>
 1      1       1 Pilot                         7.6        3706 2005-03-24
 2      1       2 Diversity Day                 8.3        3566 2005-03-29
 3      1       3 Health Care                   7.9        2983 2005-04-05
 4      1       4 The Alliance                  8.1        2886 2005-04-12
 5      1       5 Basketball                    8.4        3179 2005-04-19
 6      1       6 Hot Girl                      7.8        2852 2005-04-26
 7      2       1 The Dundies                   8.7        3213 2005-09-20
 8      2       2 Sexual Harassment             8.2        2736 2005-09-27
 9      2       3 Office Olympics               8.4        2742 2005-10-04
10      2       4 The Fire                      8.4        2713 2005-10-11
```

These data are from data.world, by way of TidyTuesday.

Rating vs Air Date



rsample

www.rstudio.com

Test-train split

```
set.seed(123)
(office_split = initial_split(office_ratings, prop = 0.8))
```

```
<Training/Testing/Total>
<150/38/188>
```

```
(office_train = training(office_split))
```

```
# A tibble: 150 × 6
   season episode title    imdb_rating total_votes
    <dbl>   <dbl> <chr>          <dbl>       <dbl>
 1      8      18 Last Da…         7.8        1429
 2      9      14 Vandali…         7.6        1402
 3      2       8 Perform…         8.2        2416
 4      9       5 Here Co…         7.1        1515
 5      3      22 Beach G…         9.1        2783
 6      7       1 Nepotism         8.4        1897
 7      3      15 Phyllis…         8.3        2283
 8      9      21 Livin' …         8.9        2041
 9      9      18 Promos           8          1445
10      8      12 Pool Pa…         8          1612
# ℹ 140 more rows
# ℹ 1 more variable: air_date <date>
```

```
(office_test = testing(office_split))
```

```
# A tibble: 38 × 6
   season episode title    imdb_rating total_votes
    <dbl>   <dbl> <chr>          <dbl>       <dbl>
 1      1       2 Diversi…         8.3        3566
 2      2       4 The Fire         8.4        2713
 3      2       9 E-Mail …         8.4        2527
 4      2      12 The Inj…         9          3282
 5      2      22 Casino …         9.3        3644
 6      3       5 Initiat…         8.2        2254
 7      3      16 Busines…         8.8        2622
 8      3      17 Cocktai…         8.5        2264
 9      4       6 Branch …         8.5        2185
10      4       7 Survivo…         8.3        2110
# ℹ 28 more rows
# ℹ 1 more variable: air_date <date>
```

Feature engineering with dplyr

```
office_train |>
  mutate(
    season = as_factor(season),
    month = lubridate::month(air_date),
    wday = lubridate::wday(air_date),
    top10_votes = as.integer(total_votes > quantile(total_votes, 0.9))
  )
```

```
# A tibble: 150 × 9
   season episode title                imdb_rating total_votes air_date   month  wday top10_votes
   <fct>    <dbl> <chr>                      <dbl>       <dbl> <date>     <dbl> <dbl>       <int>
 1 8           18 Last Day in Florida          7.8        1429 2012-03-08     3     5           0
 2 9           14 Vandalism                    7.6        1402 2013-01-31     1     5           0
 3 2            8 Performance Review           8.2        2416 2005-11-15    11     3           0
 4 9            5 Here Comes Treble            7.1        1515 2012-10-25    10     5           0
 5 3           22 Beach Games                  9.1        2783 2007-05-10     5     5           0
 6 7            1 Nepotism                     8.4        1897 2010-09-23     9     5           0
 7 3           15 Phyllis' Wedding             8.3        2283 2007-02-08     2     5           0
 8 9           21 Livin' the Dream             8.9        2041 2013-05-02     5     5           0
 9 9           18 Promos                       8          1445 2013-04-04     4     5           0
10 8           12 Pool Party                   8          1612 2012-01-19     1     5           0
# ℹ 140 more rows
```

recipes

www.rstudio.com

Better living through recipes

```r
(r = recipe(imdb_rating ~ ., data = office_train))
```

```r
summary(r)
```

```
# A tibble: 6 × 4
  variable    type      role      source
  <chr>       <list>    <chr>     <chr>
1 season      <chr [2]> predictor original
2 episode     <chr [2]> predictor original
3 title       <chr [3]> predictor original
4 total_votes <chr [2]> predictor original
5 air_date    <chr [1]> predictor original
6 imdb_rating <chr [2]> outcome   original
```

Recipe roles

```r
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
    update_role(title, new_role = "ID")
)
```

```r
summary(r)
```

```
# A tibble: 6 × 4
  variable    type      role      source
  <chr>       <list>    <chr>     <chr>
1 season      <chr [2]> predictor original
2 episode     <chr [2]> predictor original
3 title       <chr [3]> ID        original
4 total_votes <chr [2]> predictor original
5 air_date    <chr [1]> predictor original
6 imdb_rating <chr [2]> outcome   original
```

Adding features (month & day of week)

```r
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
    update_role(title, new_role = "ID") |>
    step_date(air_date, features = c("dow", "mor
)
```

```r
summary(r)
```

```
# A tibble: 6 × 4
  variable    type      role      source
  <chr>       <list>    <chr>     <chr>
1 season      <chr [2]> predictor original
2 episode     <chr [2]> predictor original
3 title       <chr [3]> ID        original
4 total_votes <chr [2]> predictor original
5 air_date    <chr [1]> predictor original
6 imdb_rating <chr [2]> outcome   original
```

Adding Holidays

```
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
  update_role(title, new_role = "ID") |>
  step_date(air_date, features = c("dow", "month")) |>
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
    keep_original_cols = FALSE
  )
)
```

Seasons as factors

```r
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
  update_role(title, new_role = "ID") |>
  step_date(air_date, features = c("dow", "month")) |>
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
    keep_original_cols = FALSE
  ) |>
  step_num2factor(season, levels = as.character(1:9))
)
```

Dummy coding

```r
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
  update_role(title, new_role = "ID") |>
  step_date(air_date, features = c("dow", "month")) |>
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
    keep_original_cols = FALSE
  ) |>
  step_num2factor(season, levels = as.character(1:9)) |>
  step_dummy(all_nominal_predictors())
)
```

top10_votes

```r
( r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
  update_role(title, new_role = "ID") |>
  step_date(air_date, features = c("dow", "month")) |>
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
    keep_original_cols = FALSE
  ) |>
  step_num2factor(season, levels = as.character(1:9)) |>
  step_dummy(all_nominal_predictors()) |>
  step_percentile(total_votes) |>
  step_mutate(top10 = as.integer(total_votes >= 0.9)) |>
  step_rm(total_votes)
)
```

Preparing a recipe

```r
prep(r)
```

Baking a recipe

```r
prep(r) |>
  bake(new_data = office_train)
```

```
# A tibble: 150 × 33
   episode title    imdb_rating air_date_USThanksgiv…¹ air_date_USChristmas…² air_date_USNewYearsDay
     <dbl> <fct>          <dbl>                  <int>                  <int>                  <int>
 1      18 Last Da…         7.8                      0                      0                      0
 2      14 Vandali…         7.6                      0                      0                      0
 3       8 Perform…         8.2                      0                      0                      0
 4       5 Here Co…         7.1                      0                      0                      0
 5      22 Beach G…         9.1                      0                      0                      0
 6       1 Nepotism         8.4                      0                      0                      0
 7      15 Phyllis…         8.3                      0                      0                      0
 8      21 Livin' …         8.9                      0                      0                      0
 9      18 Promos           8                        0                      0                      0
10      12 Pool Pa…         8                        0                      0                      0
# ℹ 140 more rows
# ℹ abbreviated names: ¹air_date_USThanksgivingDay, ²air_date_USChristmasDay
# ℹ 27 more variables: air_date_USIndependenceDay <int>, season_X2 <dbl>, season_X3 <dbl>,
#   season_X4 <dbl>, season_X5 <dbl>, season_X6 <dbl>, season_X7 <dbl>, season_X8 <dbl>,
```

Informative features?

```
prep(r) |>
  bake(new_data = office_train) |>
  map_int(~ length(unique(.x)))
```

```
                 episode                    title              imdb_rating
                      26                      150                       26
air_date_USThanksgivingDay   air_date_USChristmasDay    air_date_USNewYearsDay
                       1                        1                        1
air_date_USIndependenceDay                season_X2                season_X3
                       1                        2                        2
               season_X4                season_X5                season_X6
                       2                        2                        2
               season_X7                season_X8                season_X9
                       2                        2                        2
         air_date_dow_Mon         air_date_dow_Tue         air_date_dow_Wed
                       1                        2                        1
         air_date_dow_Thu         air_date_dow_Fri         air_date_dow_Sat
                       2                        1                        1
       air_date_month_Feb       air_date_month_Mar       air_date_month_Apr
                       2                        2                        2
       air_date_month_May       air_date_month_Jun       air_date_month_Jul
```

Removing zero variance predictors

```r
r = recipe(
    imdb_rating ~ ., data = office_train
  ) |>
  update_role(title, new_role = "ID") |>
  step_date(air_date, features = c("dow", "month")) |>
  step_holiday(
    air_date,
    holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
    keep_original_cols = FALSE
  ) |>
  step_num2factor(season, levels = as.character(1:9)) |>
  step_dummy(all_nominal_predictors()) |>
  step_percentile(total_votes) |>
  step_mutate(top10 = as.integer(total_votes >= 0.9)) |>
  step_rm(total_votes) |>
  step_zv(all_predictors())
```

```r
prep(r) |>
  bake(new_data = office_train)
```

```
# A tibble: 150 × 22
   episode title   imdb_rating season_X2 season_X3 season_X4 season_X5 season_X6 season_X7 season_X8
     <dbl> <fct>         <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
 1      18 Last D…         7.8         0         0         0         0         0         0         1
 2      14 Vandal…         7.6         0         0         0         0         0         0         0
 3       8 Perfor…         8.2         1         0         0         0         0         0         0
 4       5 Here C…         7.1         0         0         0         0         0         0         0
 5      22 Beach …         9.1         0         1         0         0         0         0         0
 6       1 Nepoti…         8.4         0         0         0         0         0         1         0
 7      15 Phylli…         8.3         0         1         0         0         0         0         0
 8      21 Livin'…         8.9         0         0         0         0         0         0         0
 9      18 Promos          8           0         0         0         0         0         0         0
10      12 Pool P…         8           0         0         0         0         0         0         1
# ℹ 140 more rows
# ℹ 12 more variables: season_X9 <dbl>, air_date_dow_Tue <dbl>, air_date_dow_Thu <dbl>,
#   air_date_month_Feb <dbl>, air_date_month_Mar <dbl>, air_date_month_Apr <dbl>,
#   air_date_month_May <dbl>, air_date_month_Sep <dbl>, air_date_month_Oct <dbl>,
```

Really putting it all together

```r
(office_work = workflow() |>
  add_recipe(r) |>
  add_model(
    linear_reg() |>
    set_engine("lm")
  )
)
```

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

── Preprocessor ────────────────────────────────────────────────────────────────
8 Recipe Steps

• step_date()
• step_holiday()
• step_num2factor()
• step_dummy()
• step_percentile()
• step_mutate()
• step_rm()
• step_zv()

── Model ───────────────────────────────────────────────────────────────────────
```

Workflow fit

```r
(office_fit = office_work |>
  fit(data = office_train))
```

```
══ Workflow [trained] ══════════════════════════════════════════════════════════
Preprocessor: Recipe
Model: linear_reg()

── Preprocessor ────────────────────────────────────────────────────────────────
8 Recipe Steps

• step_date()
• step_holiday()
• step_num2factor()
• step_dummy()
• step_percentile()
• step_mutate()
• step_rm()
• step_zv()

── Model ───────────────────────────────────────────────────────────────────────
```

Performance

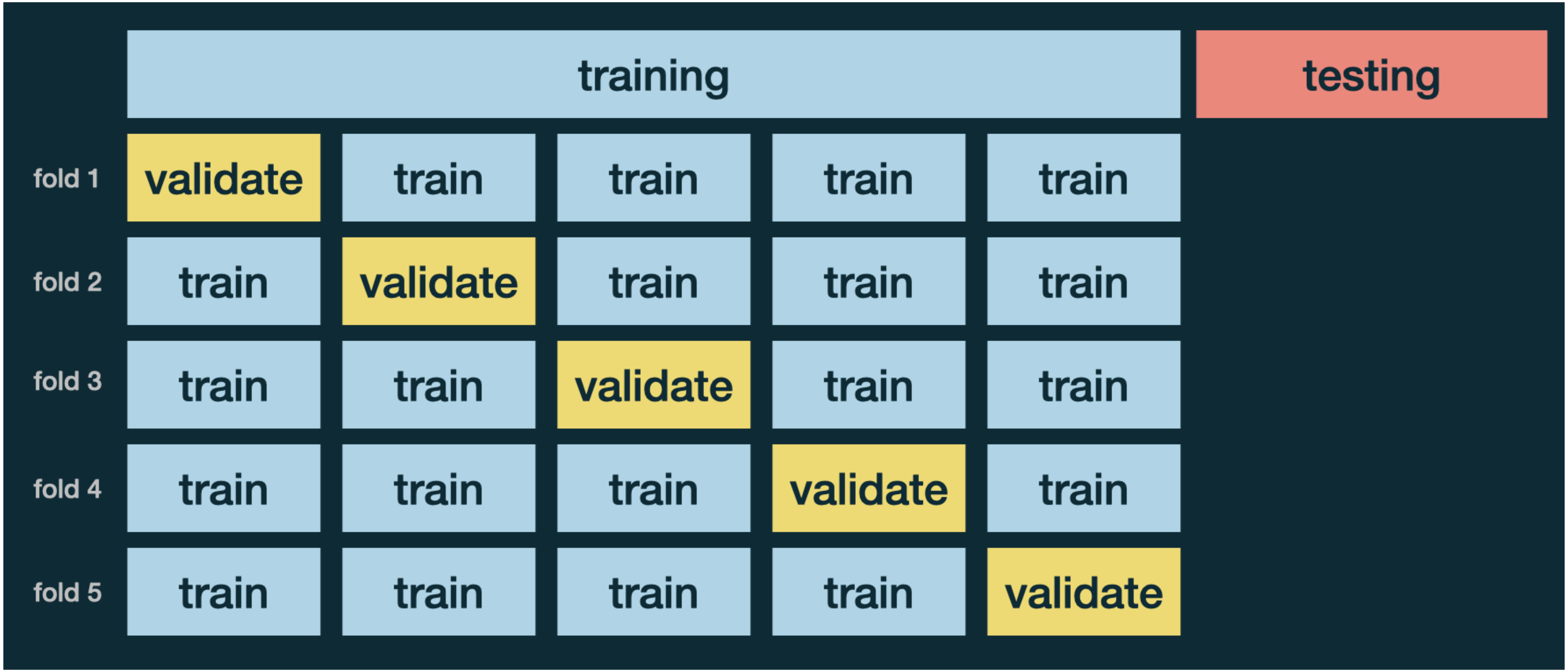
```
office_fit |>
  augment(office_train) |>
  rmse(imdb_rating, .pred)
```

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard       0.342
```

```
office_fit |>
  augment(office_test) |>
  rmse(imdb_rating, .pred)
```

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard       0.399
```

k-fold cross validation



Creating folds

```r
set.seed(123)
(folds = vfold_cv(office_train, v=5))
```

```
#  5-fold cross-validation
# A tibble: 5 × 2
  splits           id
  <list>           <chr>
1 <split [120/30]> Fold1
2 <split [120/30]> Fold2
3 <split [120/30]> Fold3
4 <split [120/30]> Fold4
5 <split [120/30]> Fold5
```

```r
(office_fit_folds = office_work |>
  fit_resamples(folds)
)
```

```
# Resampling results
# 5-fold cross-validation
# A tibble: 5 × 4
  splits           id    .metrics         .notes
  <list>           <chr> <list>           <list>
1 <split [120/30]> Fold1 <tibble [2 × 4]> <tibble [0 × 3]>
2 <split [120/30]> Fold2 <tibble [2 × 4]> <tibble [1 × 3]>
3 <split [120/30]> Fold3 <tibble [2 × 4]> <tibble [0 × 3]>
4 <split [120/30]> Fold4 <tibble [2 × 4]> <tibble [0 × 3]>
5 <split [120/30]> Fold5 <tibble [2 × 4]> <tibble [0 × 3]>
```

```
There were issues with some computations:

  - Warning(s) x1: prediction from rank-deficient fit; consider predict(., rankdeficient="NA")

Run `show_notes(.Last.tune.result)` for more information.
```

Fold performance

```r
tune::collect_metrics(office_fit_folds)
```

```
# A tibble: 2 × 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 rmse    standard   0.420     5  0.0182 Preprocessor1_Model1
2 rsq     standard   0.429     5  0.0597 Preprocessor1_Model1
```

```r
tune::collect_metrics(office_fit_folds, summarize = FALSE) |>
  filter(.metric == "rmse")
```

```
# A tibble: 5 × 5
  id    .metric .estimator .estimate .config
  <chr> <chr>   <chr>          <dbl> <chr>
1 Fold1 rmse    standard       0.467 Preprocessor1_Model1
2 Fold2 rmse    standard       0.403 Preprocessor1_Model1
3 Fold3 rmse    standard       0.405 Preprocessor1_Model1
4 Fold4 rmse    standard       0.454 Preprocessor1_Model1
5 Fold5 rmse    standard       0.368 Preprocessor1_Model1
```

More on the `tune` package next time