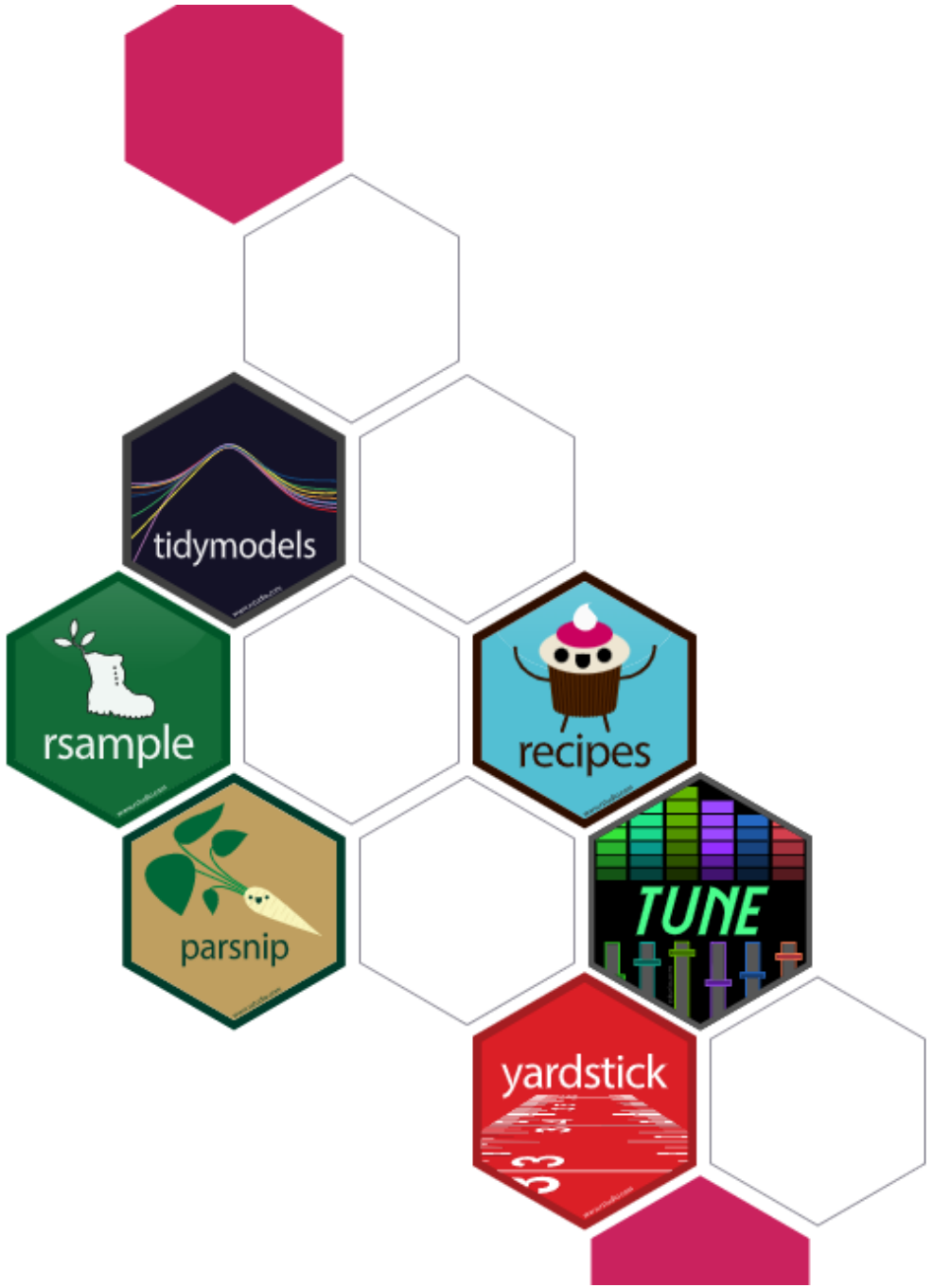
Tidymodels

Lecture 22

Dr. Colin Rundel


tidymodels
rsample
recipes
parsnip
TUNE
yardstick

Tidymodels

```
1 library(tidymodels)
```

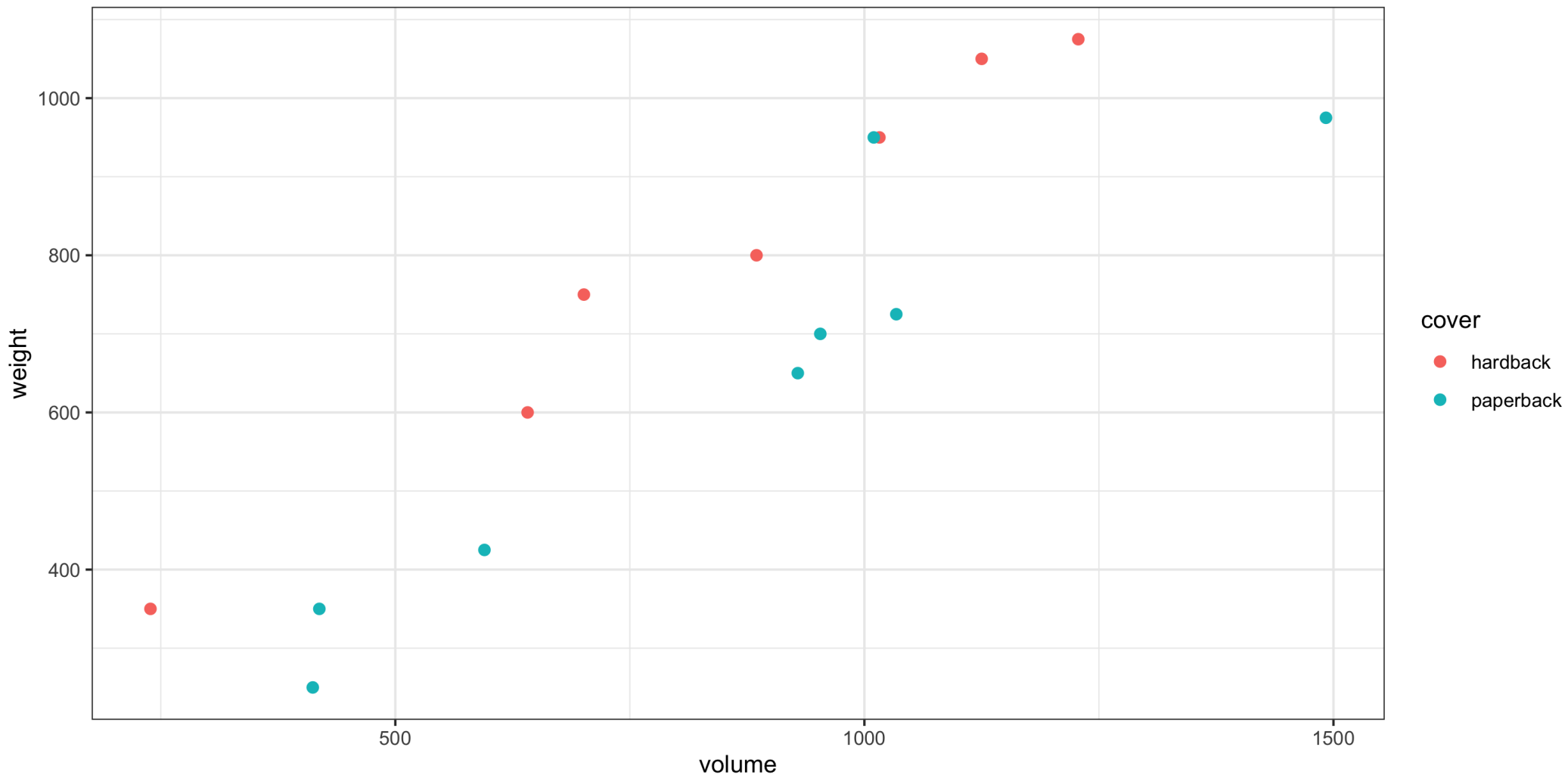
```
── Attaching packages ──────────────────── tidymodels 1.0.0 ──

✔ broom        1.0.1     ✔ rsample      1.1.0
✔ dials        1.1.0     ✔ tune         1.0.1
✔ infer        1.0.3     ✔ workflows    1.1.0
✔ modeldata    1.0.1     ✔ workflowsets 1.0.0
✔ parsnip      1.0.3     ✔ yardstick    1.1.0
✔ recipes      1.0.3

── 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()
✖ recipes::step()     masks stats::step()
• Learn how to get started at https://www.tidymodels.org/start/
```

Book data

```
(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
```

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



Building a tidymodel

```
1 linear_reg()
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Building a tidymodel

```
1 linear_reg() %>%
2   set_engine("lm")
```

```
Linear Regression Model Specification (regression)

Computational engine: lm
```

Building a tidymodel

```
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
            161.58654                0.76159
      coverpaperback  volume:coverpaperback
          -120.21407               -0.07573
```

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

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

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


broom
www.tidyverse.org

Tidy model objects

```
lm(weight ~ volume * cover, data = books) %>%
  summary()
```

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

Residuals:
   Min     1Q Median     3Q    Max
-89.67 -32.07 -21.82  17.94 215.91

Coefficients:
                       Estimate Std. Error t value
(Intercept)           161.58654   86.51918   1.868
volume                  0.76159    0.09718   7.837
coverpaperback       -120.21407  115.65899  -1.039
volume:coverpaperback   -0.07573    0.12802  -0.592
                      Pr(>|t|)
(Intercept)             0.0887 .
volume                7.94e-06 ***
```

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

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

```
# A tibble: 4 × 5
  term                  estimate std.error statis…¹ p.value
  <chr>                    <dbl>     <dbl>    <dbl>   <dbl>
1 (Intercept)             162.       86.5      1.87  8.87e-2
2 volume                    0.762     0.0972   7.84  7.94e-6
3 coverpaperback         -120.      116.      -1.04  3.21e-1
4 volume:coverpaperback    -0.0757    0.128   -0.592 5.66e-1
# … with abbreviated variable name ¹statistic
```

Tidy model statistics

```
broom::glance(lm(weight ~ volume * cover, data = books))
```

```
# A tibble: 1 × 12
  r.squared adj.r…¹ sigma stati…² p.value    df logLik   AIC
      <dbl>   <dbl> <dbl>   <dbl>   <dbl> <dbl>  <dbl> <dbl>
1     0.930   0.911  80.4    48.5 1.24e-6     3  -84.8  180.
# … with 4 more variables: BIC <dbl>, deviance <dbl>,
#   df.residual <int>, nobs <int>, and abbreviated variable
#   names ¹adj.r.squared, ²statistic
```

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

```
# A tibble: 1 × 12
  r.squared adj.r…¹ sigma stati…² p.value    df logLik   AIC
      <dbl>   <dbl> <dbl>   <dbl>   <dbl> <dbl>  <dbl> <dbl>
1     0.930   0.911  80.4    48.5 1.24e-6     3  -84.8  180.
# … with 4 more variables: BIC <dbl>, deviance <dbl>,
#   df.residual <int>, nobs <int>, and abbreviated variable
#   names ¹adj.r.squared, ²statistic
```

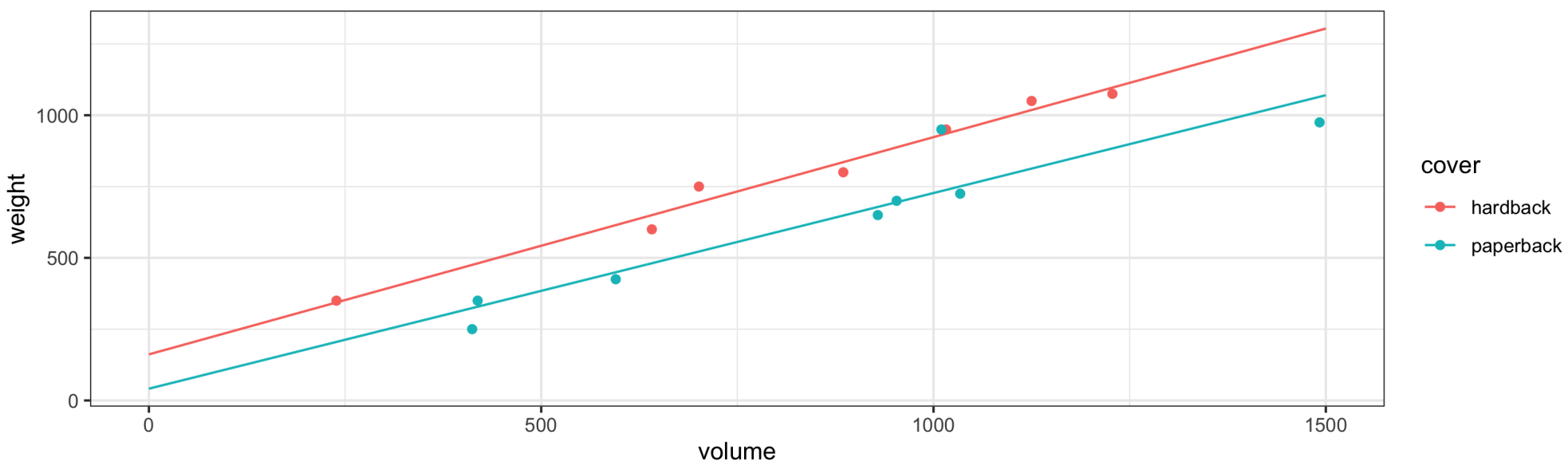
Tidy model prediction

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

```
# A tibble: 15 × 5
   volume weight cover     .pred .resid
    <dbl>  <dbl> <fct>     <dbl>  <dbl>
 1    885    800 hardback   836. -35.6
 2   1016    950 hardback   935.  14.6
 3   1125   1050 hardback  1018.  31.6
 4    239    350 hardback   344.   6.39
 5    701    750 hardback   695.  54.5
 6    641    600 hardback   650. -49.8
 7   1228   1075 hardback  1097. -21.8
 8    412    250 paperback  324. -73.9
 9    953    700 paperback  695.   5.00
10    929    650 paperback  679.  28.5
```

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?

```
1 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
```

```
1 (bayes_tm = linear_reg() %>%
2   set_engine(
3     "stan",
4     prior_intercept = rstanarm::student_t(df = 1
5     prior = rstanarm::student_t(df = 1),
6     seed = 1234
7   )
8 )
```

```
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)          91.1   63.0
volume                0.8    0.1
coverpaperback        0.1    3.7
volume:coverpaperback -0.2    0.1

Auxiliary parameter(s):
      Median MAD_SD
sigma 89.3   19.0
```

See ?details_linear_reg_stan for details within parsnip

What was actually run?

```
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)             91.1      63.0
2 volume                   0.831     0.0765
3 coverpaperback           0.0550    3.66
4 volume:coverpaperback   -0.198     0.0537
```

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

```
# A tibble: 1 × 4
  algorithm   pss  nobs sigma
  <chr>     <dbl> <int> <dbl>
1 sampling   4000    15  89.3
```

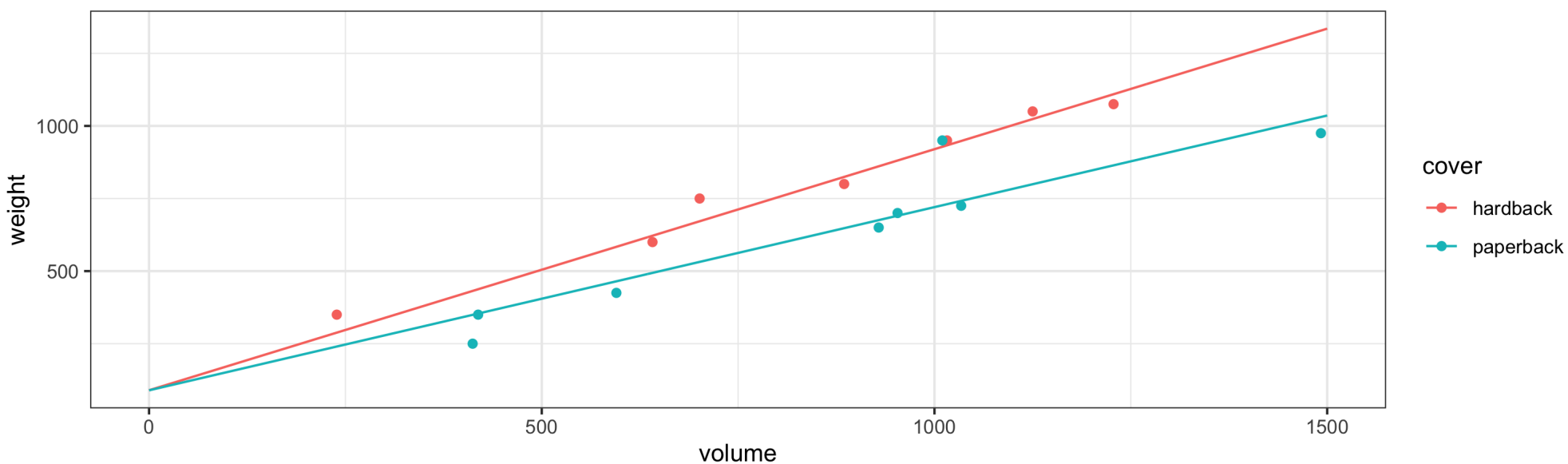
Augment

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

```
# A tibble: 15 × 5
   volume weight cover     .pred .resid
    <dbl>  <dbl> <fct>     <dbl>  <dbl>
 1    885    800 hardback   824. -24.4
 2   1016    950 hardback   933.  16.8
 3   1125   1050 hardback  1024.  26.3
 4    239    350 hardback   288.  61.9
 5    701    750 hardback   672.  78.4
 6    641    600 hardback   622. -21.8
 7   1228   1075 hardback  1109. -34.2
 8    412    250 paperback  349. -99.3
 9    953    700 paperback  691.   9.31
10    929    650 paperback  676.  25.5
```

Predictions

```
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)
```




yardstick
www.rstudio.com

Performance

```
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
```

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

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard        72.4
```

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

Cross validation and Feature engineering

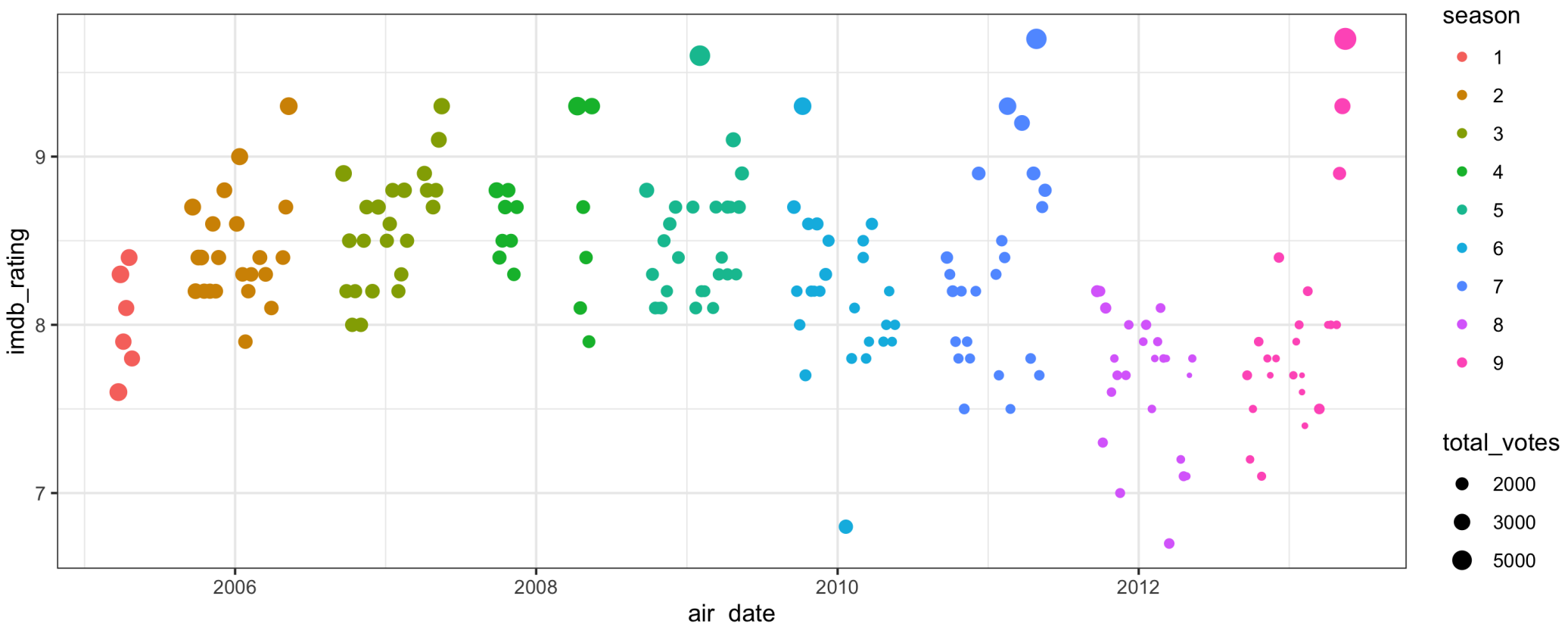
The Office & IMDB

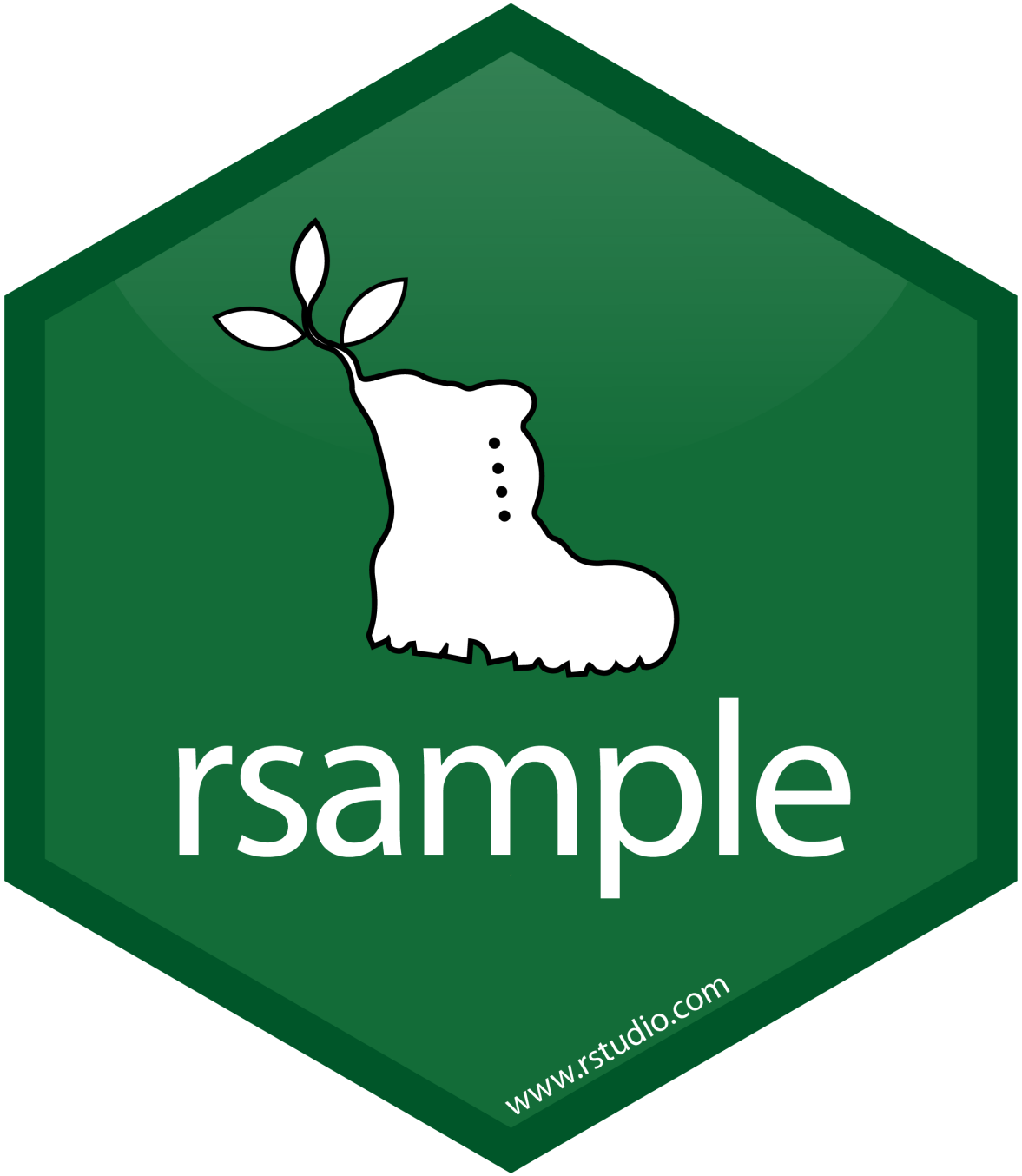
```
1 (office_ratings = read_csv("data/office_ratings.csv"))
```

```
# A tibble: 188 × 6
   season episode title           imdb_…¹ total…² 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 Harassm…     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

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

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

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

```
# A tibble: 150 × 6
   season episode title imdb_…¹ total…² air_date
    <dbl>   <dbl> <chr>   <dbl>   <dbl> <date>
 1      8      18 Last…     7.8    1429 2012-03-08
 2      9      14 Vand…     7.6    1402 2013-01-31
 3      2       8 Perf…     8.2    2416 2005-11-15
 4      9       5 Here…     7.1    1515 2012-10-25
 5      3      22 Beac…     9.1    2783 2007-05-10
 6      7       1 Nepo…     8.4    1897 2010-09-23
 7      3      15 Phyl…     8.3    2283 2007-02-08
 8      9      21 Livi…     8.9    2041 2013-05-02
 9      9      18 Prom…     8      1445 2013-04-04
10      8      12 Pool…     8      1612 2012-01-19
# … with 140 more rows, and abbreviated variable
#   names ¹imdb_rating, ²total_votes
```

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

```
# A tibble: 38 × 6
   season episode title imdb_…¹ total…² air_date
    <dbl>   <dbl> <chr>   <dbl>   <dbl> <date>
 1      1       2 Dive…     8.3    3566 2005-03-29
 2      2       4 The …     8.4    2713 2005-10-11
 3      2       9 E-Ma…     8.4    2527 2005-11-22
 4      2      12 The …     9      3282 2006-01-12
 5      2      22 Casi…     9.3    3644 2006-05-11
 6      3       5 Init…     8.2    2254 2006-10-19
 7      3      16 Busi…     8.8    2622 2007-02-15
 8      3      17 Cock…     8.5    2264 2007-02-22
 9      4       6 Bran…     8.5    2185 2007-11-01
10      4       7 Surv…     8.3    2110 2007-11-08
# … with 28 more rows, and abbreviated variable
#   names ¹imdb_rating, ²total_votes
```

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
# … with 140 more rows
```


recipes
www.rstudio.com

Better living through recipes

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

```
Recipe

Inputs:

      role #variables
   outcome          1
 predictor          5
```

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

```
Recipe

Inputs:

      role #variables
        ID          1
   outcome          1
 predictor          4
```

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

```
Recipe

Inputs:

      role #variables
        ID           1
   outcome           1
 predictor           4

Operations:

Date features from air_date
```

```
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
  )
)
```

```
Recipe

Inputs:

      role #variables
        ID          1
   outcome          1
 predictor          4

Operations:

Date features from air_date
Holiday features from air_date
```

Seasons as factors

```
(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))
)
```

```
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 coding

```
(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())
)
```

```
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()
```

top10_votes

```
(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)
)
```

```
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()
Percentile transformation on total_votes
```

Preparing a recipe

```
1 prep(r)
```

```
Recipe

Inputs:

      role #variables
        ID           1
   outcome           1
 predictor           4

Training data contained 150 data points and no missing data.

Operations:

Date features from air_date [trained]
Holiday features from air_date [trained]
Factor variables from season [trained]
Dummy variables from season, air_date_dow, air_date_month [trained]
```

Baking a recipe

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

```
# A tibble: 150 × 33
   episode title     imdb_…¹ air_d…² air_d…³ air_d…⁴ air_d…⁵ seaso…⁶ seaso…⁷ seaso…⁸ seaso…⁹ seaso…ˣ
     <dbl> <fct>       <dbl>   <int>   <int>   <int>   <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1      18 Last Day…     7.8       0       0       0       0       0       0       0       0       0
 2      14 Vandalism     7.6       0       0       0       0       0       0       0       0       0
 3       8 Performa…     8.2       0       0       0       0       1       0       0       0       0
 4       5 Here Com…     7.1       0       0       0       0       0       0       0       0       0
 5      22 Beach Ga…     9.1       0       0       0       0       0       1       0       0       0
 6       1 Nepotism      8.4       0       0       0       0       0       0       0       0       0
 7      15 Phyllis'…     8.3       0       0       0       0       0       1       0       0       0
 8      21 Livin' t…     8.9       0       0       0       0       0       0       0       0       0
 9      18 Promos        8         0       0       0       0       0       0       0       0       0
10      12 Pool Par…     8         0       0       0       0       0       0       0       0       0
# … with 140 more rows, 21 more variables: season_X7 <dbl>, season_X8 <dbl>, season_X9 <dbl>,
#   air_date_dow_Mon <dbl>, air_date_dow_Tue <dbl>, air_date_dow_Wed <dbl>, air_date_dow_Thu <dbl>,
#   air_date_dow_Fri <dbl>, air_date_dow_Sat <dbl>, air_date_month_Feb <dbl>,
#   air_date_month_Mar <dbl>, air_date_month_Apr <dbl>, air_date_month_May <dbl>,
```

Informative features?

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

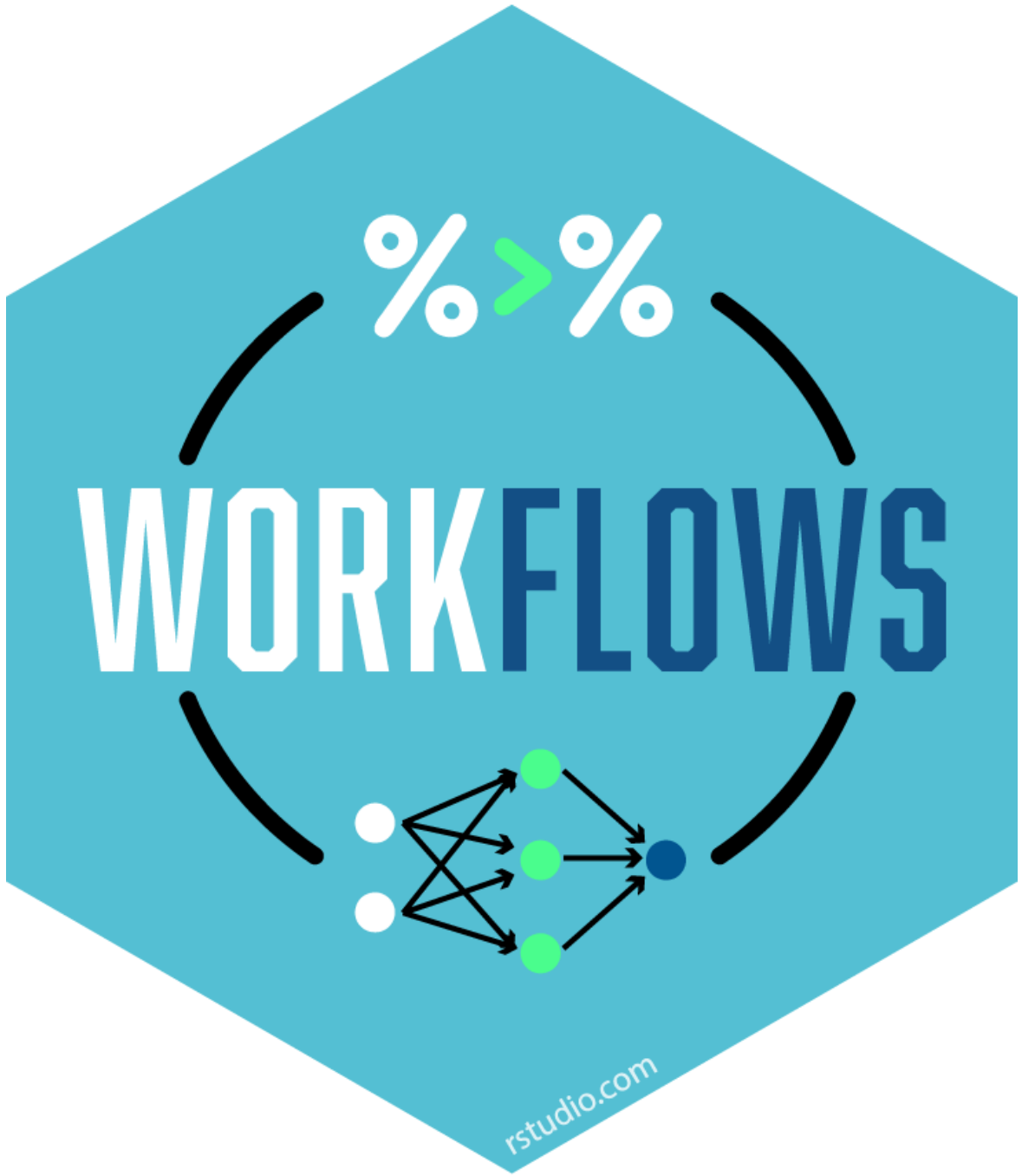
episode	title	imdb_rating
26	150	26
air_date_USThanksgivingDay	air_date_USChristmasDay	air_date_USNewYearsDay
1	1	1
air_date_USIndependenceDay	season_X2	season_X3
1	2	2
season_X4	season_X5	season_X6
2	2	2
season_X7	season_X8	season_X9
2	2	2
air_date_dow_Mon	air_date_dow_Tue	air_date_dow_Wed
1	2	1
air_date_dow_Thu	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 = 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_…¹ seaso…² seaso…³ seaso…⁴ seaso…⁵ seaso…⁶ seaso…⁷ seaso…⁸ seaso…⁹ air_d…×
     <dbl> <fct>       <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
 1      18 Last Day…     7.8       0       0       0       0       0       0       1       0       0
 2      14 Vandalism     7.6       0       0       0       0       0       0       0       1       0
 3       8 Performa…     8.2       1       0       0       0       0       0       0       0       1
 4       5 Here Com…     7.1       0       0       0       0       0       0       0       1       0
 5      22 Beach Ga…     9.1       0       1       0       0       0       0       0       0       0
 6       1 Nepotism      8.4       0       0       0       0       0       1       0       0       0
 7      15 Phyllis'…     8.3       0       1       0       0       0       0       0       0       0
 8      21 Livin' t…     8.9       0       0       0       0       0       0       0       1       0
 9      18 Promos        8         0       0       0       0       0       0       0       1       0
10      12 Pool Par…     8         0       0       0       0       0       0       1       0       0
# … with 140 more rows, 10 more variables: 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>, air_date_month_Nov <dbl>,
#   air_date_month_Dec <dbl>, top10 <int>, and abbreviated variable names ¹imdb_rating, ²season_X2,
```


%>%
WORKFLOWS
rstudio.com

Really putting it all together

```
(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

```
(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

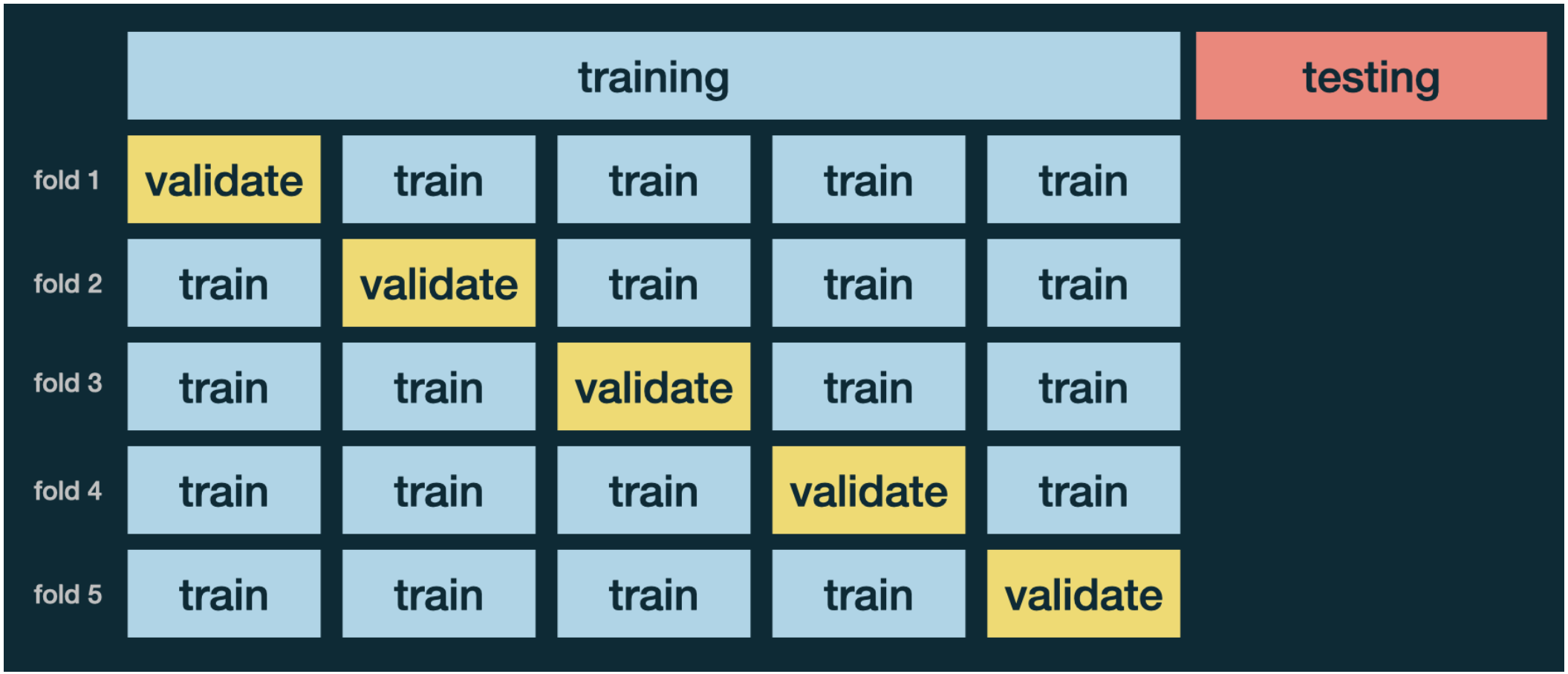
```
office_fit %>%
  augment(office_train) %>%
  rmse(imdb_rating, .pred)
```

```
# 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 a rank-deficient fit may be misleading

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

Fold performance

```
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
```

```
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