Tidymodels

Lecture 22

Dr. Colin Rundel
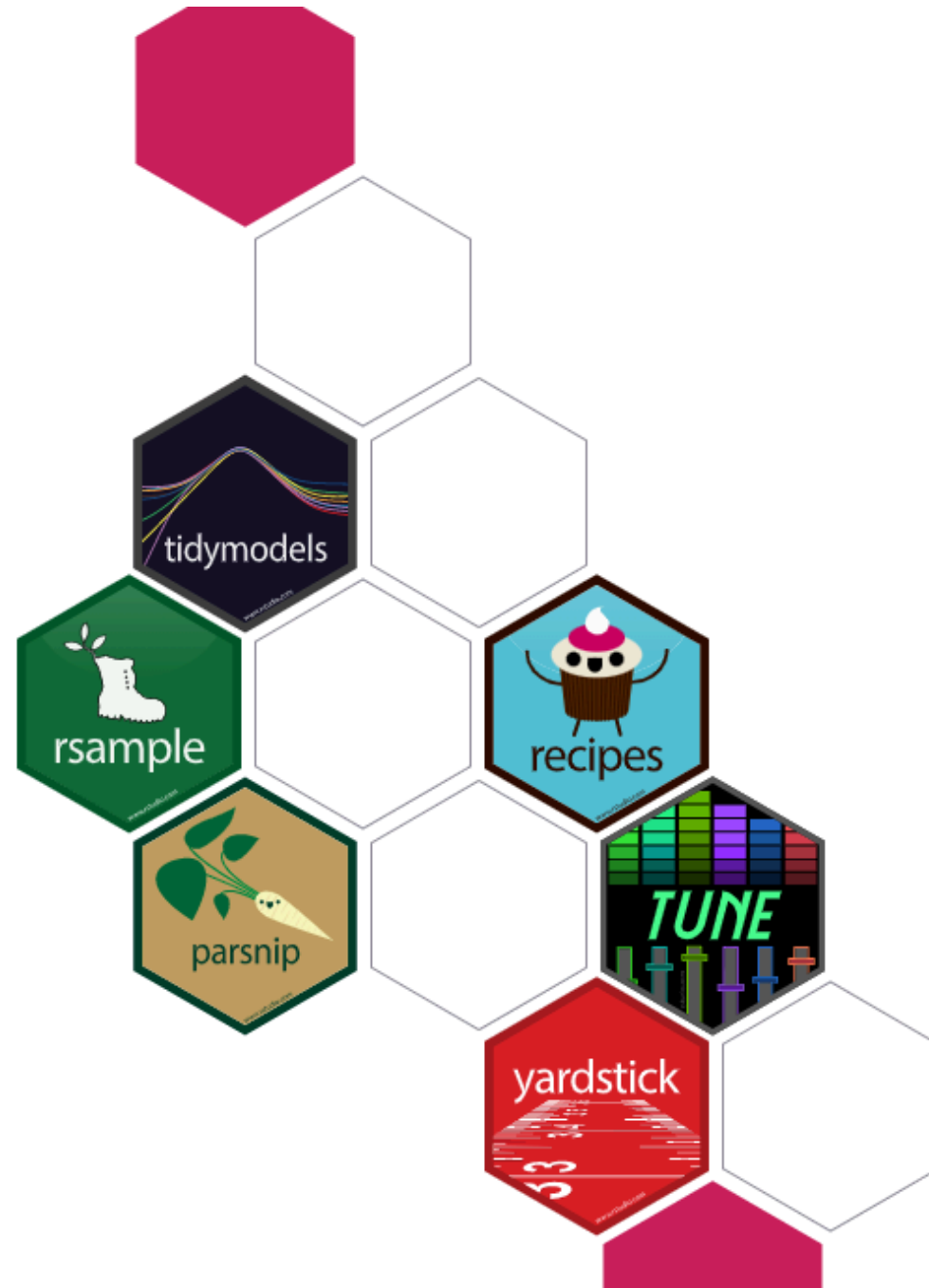
Tidymodels

```
1  library(tidymodels)
```

```
── Attaching packages ──────────────────────────────────────── tidymodels 1.2.0 ──
✔ broom        1.0.5     ✔ rsample      1.2.1
✔ dials        1.2.1     ✔ tune         1.2.0
✔ infer        1.0.7     ✔ workflows    1.1.4
✔ modeldata    1.3.0     ✔ workflowsets 1.1.0
✔ parsnip      1.2.1     ✔ yardstick    1.3.1
✔ recipes      1.0.10

── 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()
• Use tidymodels_prefer() to resolve common conflicts.
```

Book data

```r
1  (books = DAAG::allbacks |>
2    as_tibble() |>
3    select(-area) |>
4    mutate(
5      cover = forcats::fct_re
6        cover,
7        "hardback" = "hb",
8        "paperback" = "pb"
9      )
10   )
11 )
```

```
# 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)
```
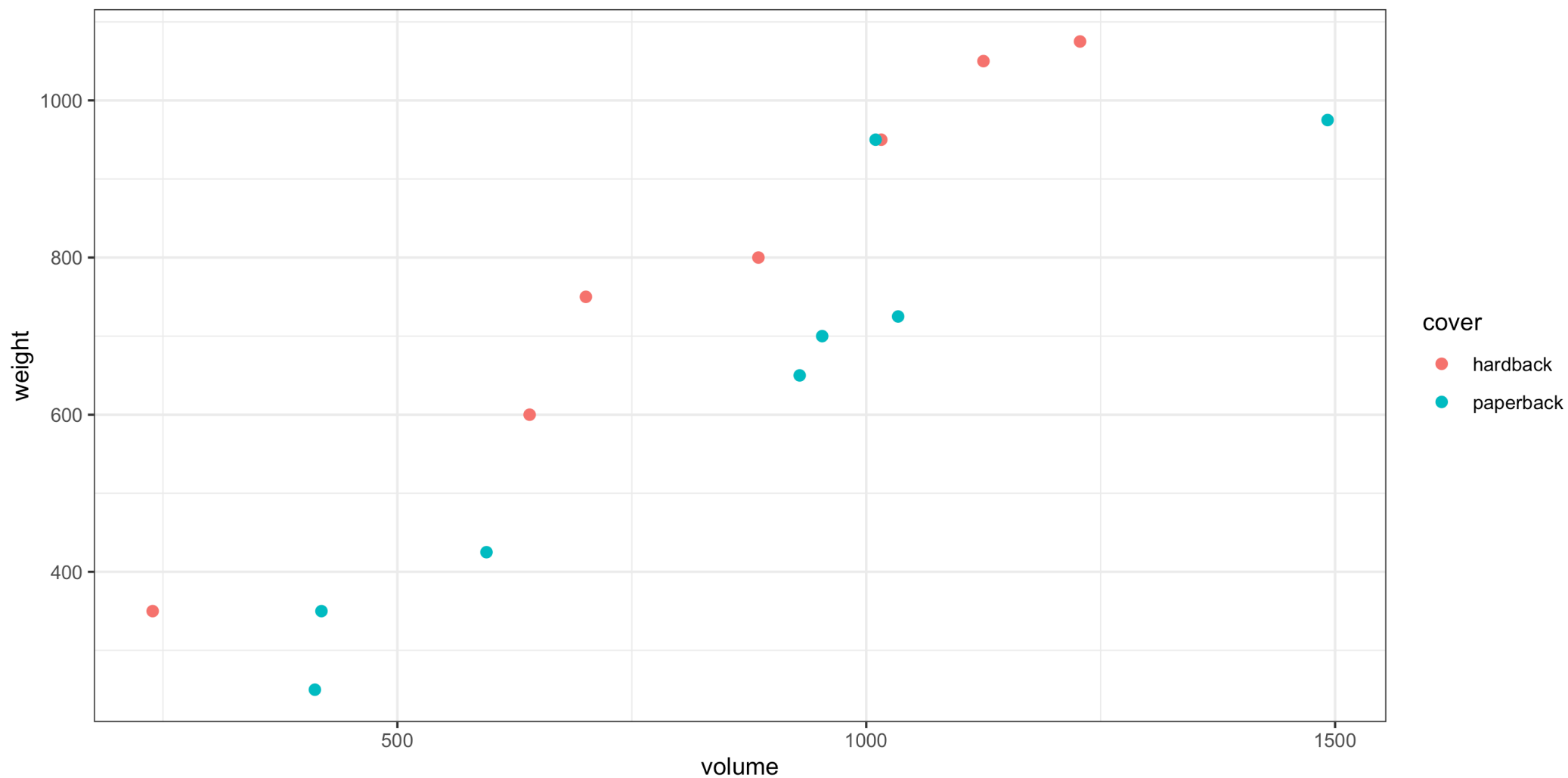
Building a tidymodel

```
1  linear_reg()
```

Linear Regression Model Specification (regression)

Computational engine: lm

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

Linear Regression Model Specification (regression)

Computational engine: lm

Building a tidymodel

```
1 linear_reg() |>
2   set_engine("lm") |>
3   fit(weight ~ volume * cover, data = bo
```

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

```
1 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

Tidy model objects

```
1  lm_tm = linear_reg() |>
2    set_engine("lm") |>
3    fit(weight ~ volume * cover, data = bo
```

```
1  lm_b = lm(weight ~ volume * cover, data
```

```
1  summary(lm_tm)
```

```
1  summary(lm_b)
```

	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

```
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 ***
```

Tidy coefficients

```
1  broom::tidy(lm_tm)
```

```
# A tibble: 4 × 5
  term          estimate std.error statistic p.value
  <chr>            <dbl>     <dbl>     <dbl>   <dbl>
1 (Intercept)    1.62e+2     86.5      1.87  8.87e-2
2 volume         7.62e-1      0.0972   7.84  7.94e-6
3 coverpaper…   -1.20e+2    116.      -1.04  3.21e-1
4 volume:cov…   -7.57e-2      0.128   -0.592 5.66e-1
```

```
1  broom::tidy(lm_b)
```

```
# A tibble: 4 × 5
  term          estimate std.error statistic p.value
  <chr>            <dbl>     <dbl>     <dbl>   <dbl>
1 (Intercept)    1.62e+2     86.5      1.87  8.87e-2
2 volume         7.62e-1      0.0972   7.84  7.94e-6
3 coverpaper…   -1.20e+2    116.      -1.04  3.21e-1
4 volume:cov…   -7.57e-2      0.128   -0.592 5.66e-1
```

Tidy statistics

```
1  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>
```

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

```
# 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 prediction

```r
1 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
10   679.   28.5    929    650 paperback
```

Putting it together

```r
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)
```
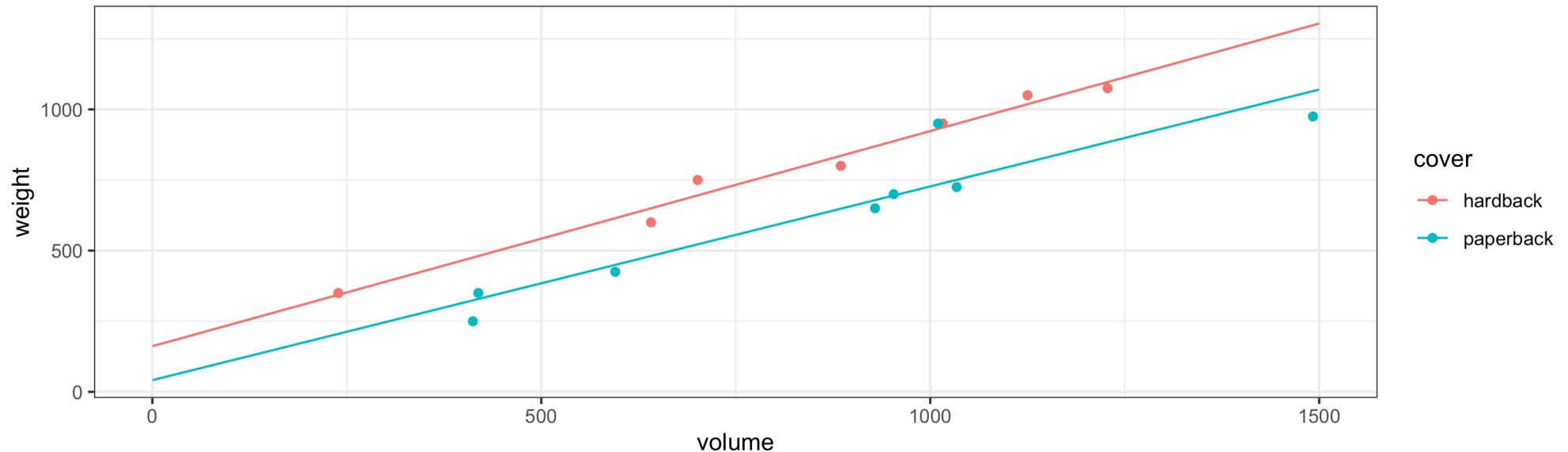
Why do we care?

```
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_
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`

```
1  (bayes_tm = bayes_tm |>
2    fit(weight ~ volume * cover, data = books)
3  )
```

```
parsnip model object

stan_glm
 family:       gaussian [identity]
 formula:      weight ~ volume * cover
 observations: 15
 predictors:   4
------
                    Median MAD_SD
(Intercept)          95.4   63.9
volume                0.8    0.1
coverpaperback       -0.3    3.6
volume:coverpaperback -0.2    0.1

Auxiliary parameter(s):
     Median MAD_SD
sigma 85.5   18.1
```

What was actually run?

```r
1  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    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

```
1  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.

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

```
# A tibble: 4 × 3
  term                  estimate std.error
  <chr>                    <dbl>     <dbl>
1 (Intercept)              95.4      63.9
2 volume                    0.828     0.0759
3 coverpaperback           -0.263     3.63
4 volume:coverpaperback    -0.197     0.0518
```

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

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

Augment

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

```
# A tibble: 15 × 5
   .pred  .resid volume weight cover
   <dbl>   <dbl>  <dbl>  <dbl> <fct>
 1  829.   -28.6    885    800 hardback
 2  937.    12.9   1016    950 hardback
 3 1027.    22.6   1125   1050 hardback
 4  294.    56.3    239    350 hardback
 5  676.    73.7    701    750 hardback
 6  627.   -26.6    641    600 hardback
 7 1113.   -37.7   1228   1075 hardback
 8  353.  -103.     412    250 paperback
 9  696.     4.34   953    700 paperback
10  680.   -30.5    929    650 paperback
11 1037.   -61.6   1492    975 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)
```
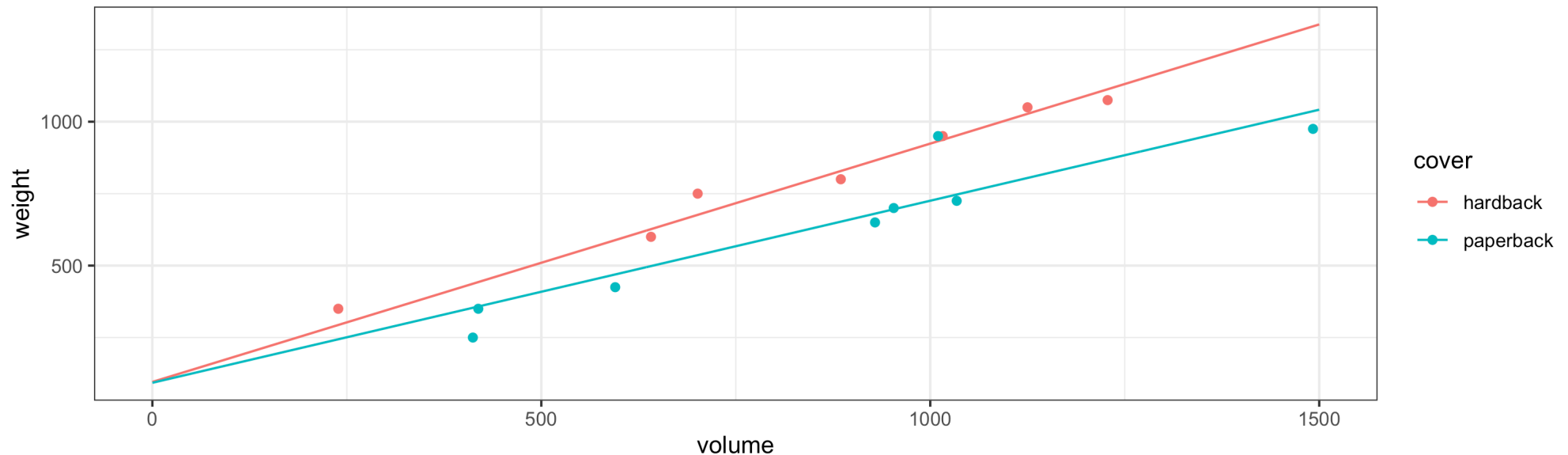
Performance

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

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

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

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard        72.0
```

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_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
# i 178 more rows
```

Rating vs Air Date
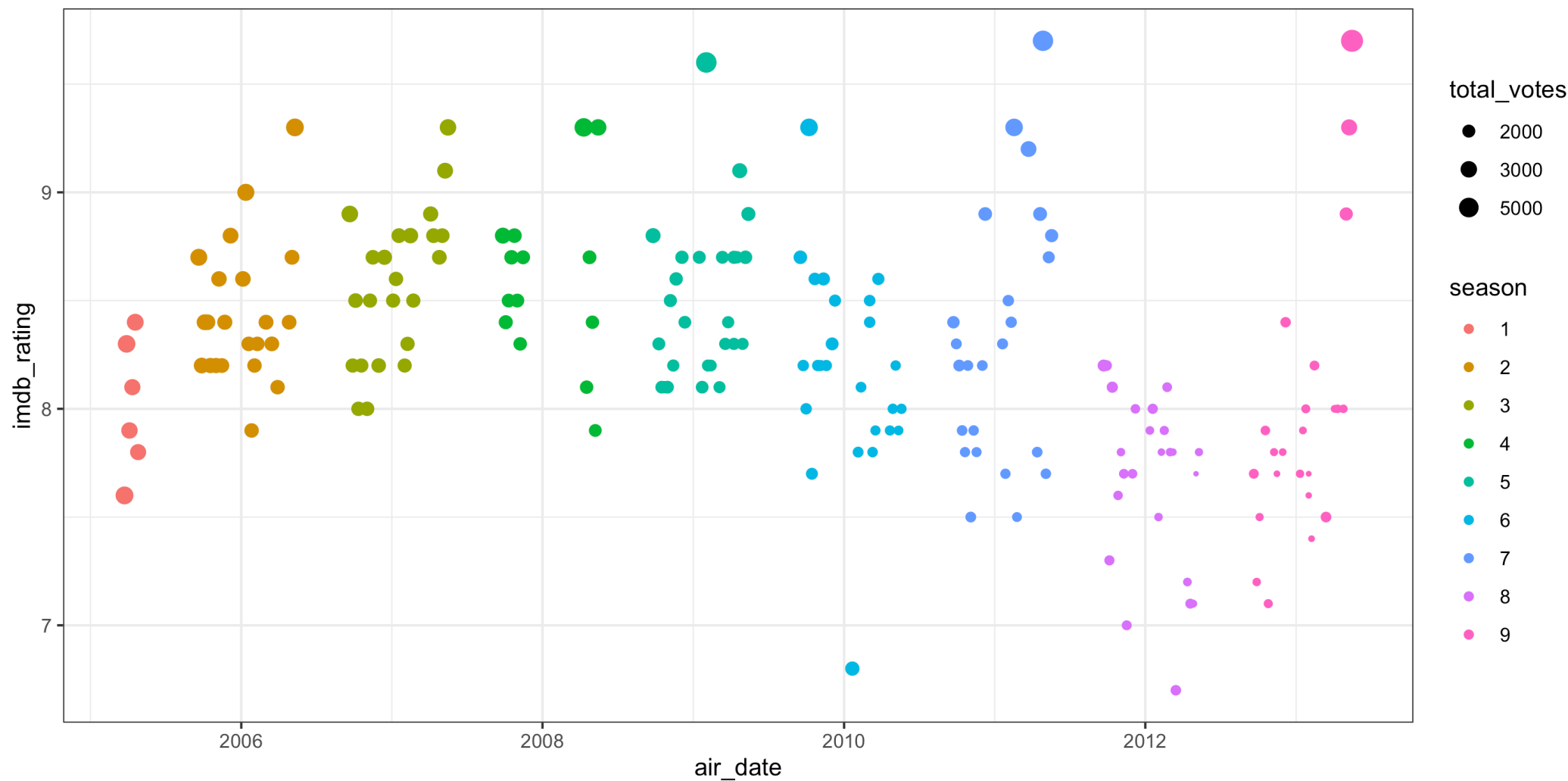
Test-train split

```
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_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>
```

```
1  (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

```
1  office_train |>
2    mutate(
3      season = as_factor(season),
4      month = lubridate::month(air_date),
5      wday = lubridate::wday(air_date),
6      top10_votes = as.integer(total_votes > quantile(total_votes, 0.9))
7    )
```

```
# 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
```

Anyone see a potential problem with the code above?

Better living through recipes

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

```
1  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
1  r = recipe(
2    imdb_rating ~ ., data = office_train
3  ) |>
4    update_role(title, new_role = "ID")
```

```r
1  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)

```
1  r = recipe(
2    imdb_rating ~ ., data = office_train
3  ) |>
4    update_role(title, new_role = "ID") |>
5    step_date(air_date, features = c("dow", "month"))
```

```
1  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
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
  )
```

```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
```

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))
```

```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
```

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())
```

```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
```

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)
```

```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
```

Preparing a recipe

```
1  prep(r)
```

── Recipe ──

── Inputs

Number of variables by role

outcome: 1
predictor: 4
ID: 1

── Training information

Training data contained 150 data points and no incomplete rows.

── 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

• Percentile transformation on: total_votes | Trained

• Variable mutation for: ~as.integer(total_votes >= 0.9) | Trained

• Variables removed: total_votes | Trained

Baking a recipe

```r
1  prep(r) |>
2    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?

```
1  prep(r) |>
2    bake(new_data = office_train) |>
3    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
1  r = recipe(
2     imdb_rating ~ ., data = office_train
3   ) |>
4   update_role(title, new_role = "ID") |>
5   step_date(air_date, features = c("dow", "month")) |>
6   step_holiday(
7     air_date,
8     holidays = c("USThanksgivingDay", "USChristmasDay", "USNewYearsDay", "USIndependenceDay"),
9     keep_original_cols = FALSE
10  ) |>
11  step_num2factor(season, levels = as.character(1:9)) |>
12  step_dummy(all_nominal_predictors()) |>
13  step_percentile(total_votes) |>
14  step_mutate(top10 = as.integer(total_votes >= 0.9)) |>
15  step_rm(total_votes) |>
16  step_zv(all_predictors())
```

```
1  prep(r) |>
2    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>,
```

Sta 323 - Spring 2024
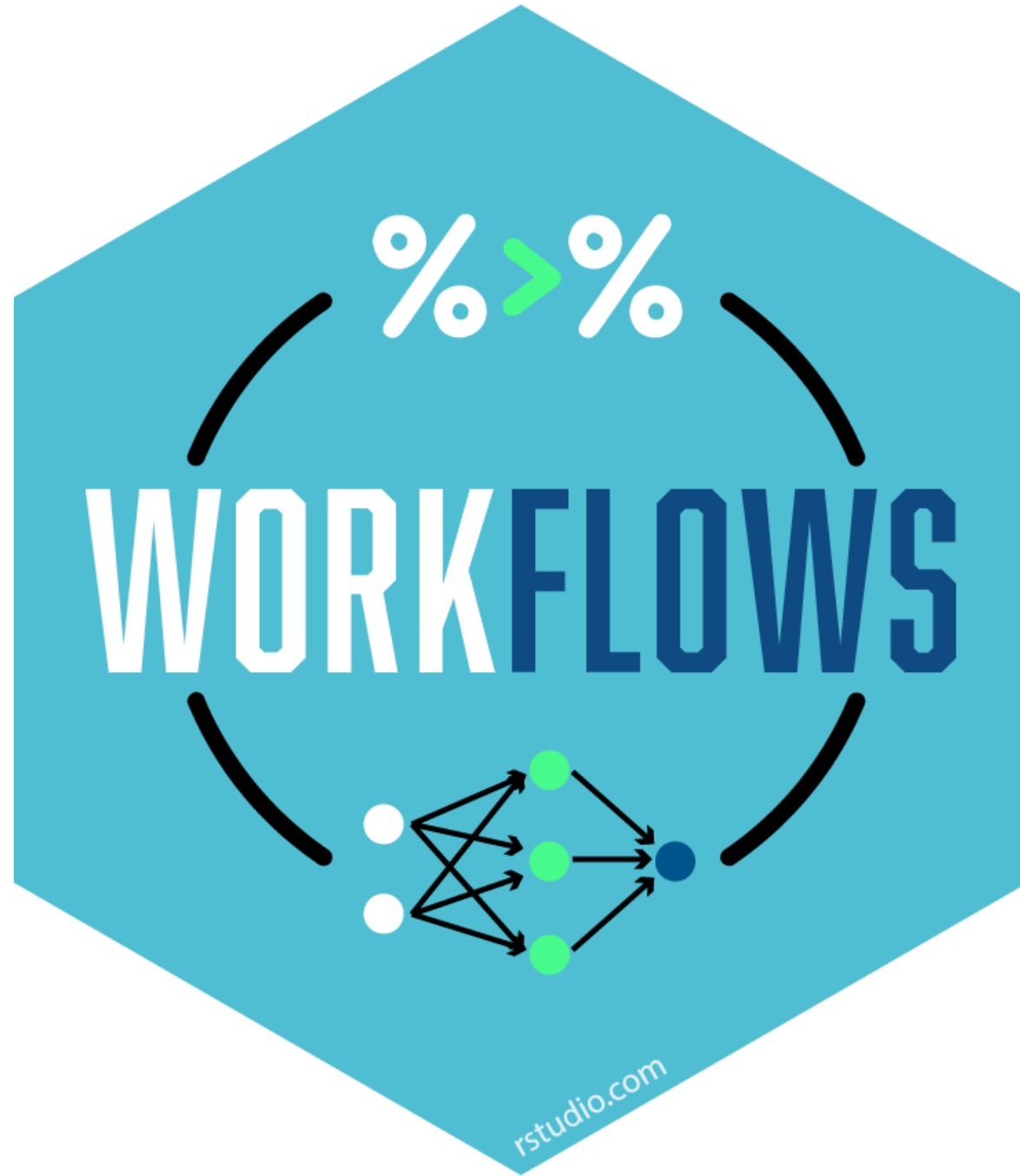
Really putting it all together

```
1  (office_work = workflow() |>
2    add_recipe(r) |>
3    add_model(
4      linear_reg() |>
5      set_engine("lm")
6    )
7  )
```

```
══ 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

```
1  (office_fit = office_work |>
2    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

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

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

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

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

k-fold cross validation

	training					testing
fold 1	validate	train	train	train	train	
fold 2	train	validate	train	train	train	
fold 3	train	train	validate	train	train	
fold 4	train	train	train	validate	train	
fold 5	train	train	train	train	validate	

Creating folds

```
1  set.seed(123)
2  (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
```

```
1  (office_fit_folds = office_work |>
2    fit_resamples(folds)
3  )
```

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

Fold performance

```r
1  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
1  tune::collect_metrics(office_fit_folds, summarize = FALSE) |>
2    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