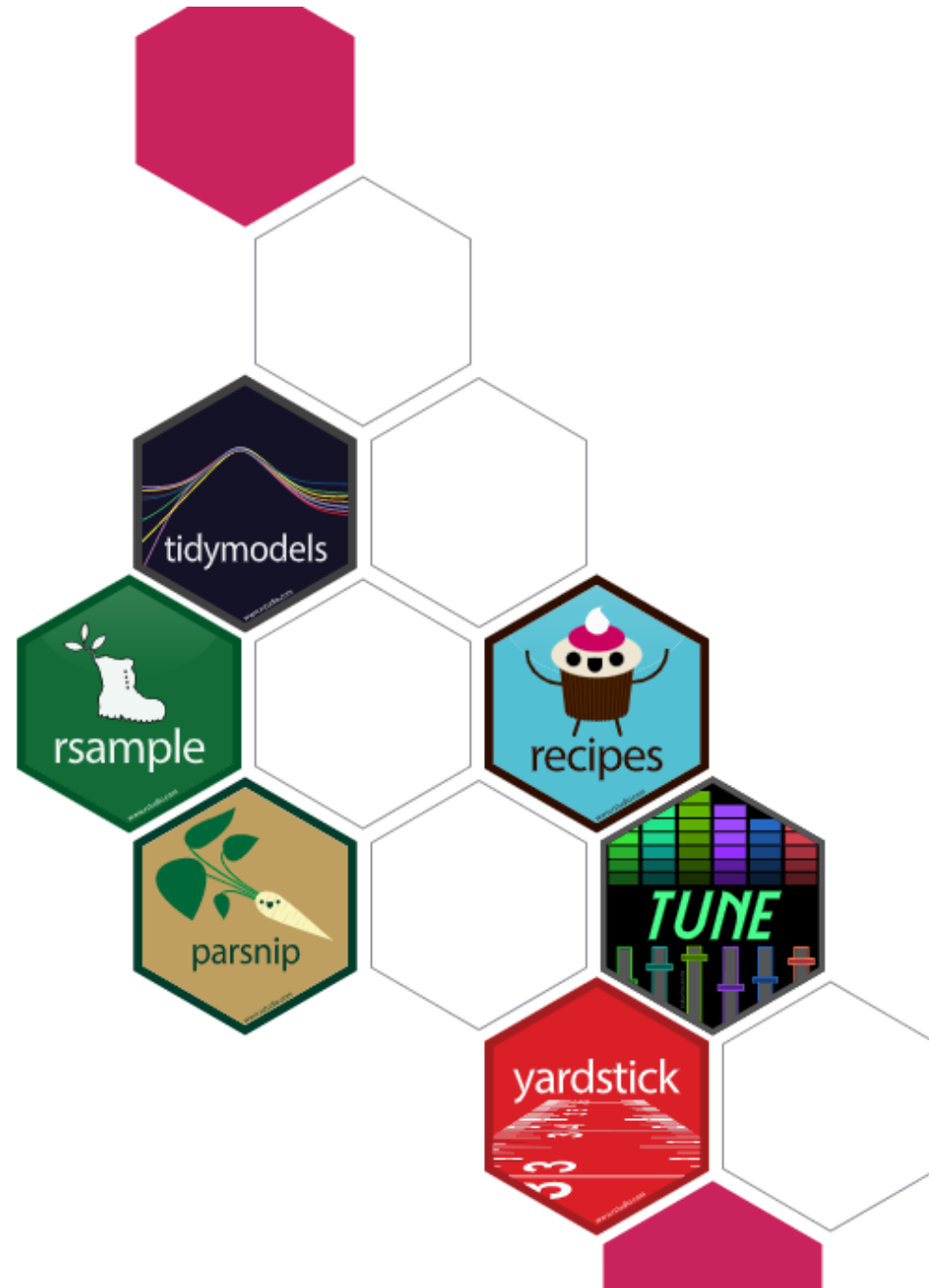


Tidymodels

Lecture 23

Dr. Colin Rundel



Tidymodels

```
1 library(tidymodels)
```

— Attaching packages — tidymodels 1.4.1 —

✓ broom	1.0.10	✓ rsample	1.3.1
✓ dials	1.4.2	✓ tailor	0.1.0
✓ infer	1.0.9	✓ tune	2.0.1
✓ modeldata	1.5.1	✓ workflows	1.3.0
✓ parsnip	1.3.3	✓ workflowsets	1.1.1
✓ recipes	1.3.1	✓ yardstick	1.3.2

— Conflicts — tidymodels_conflicts() —

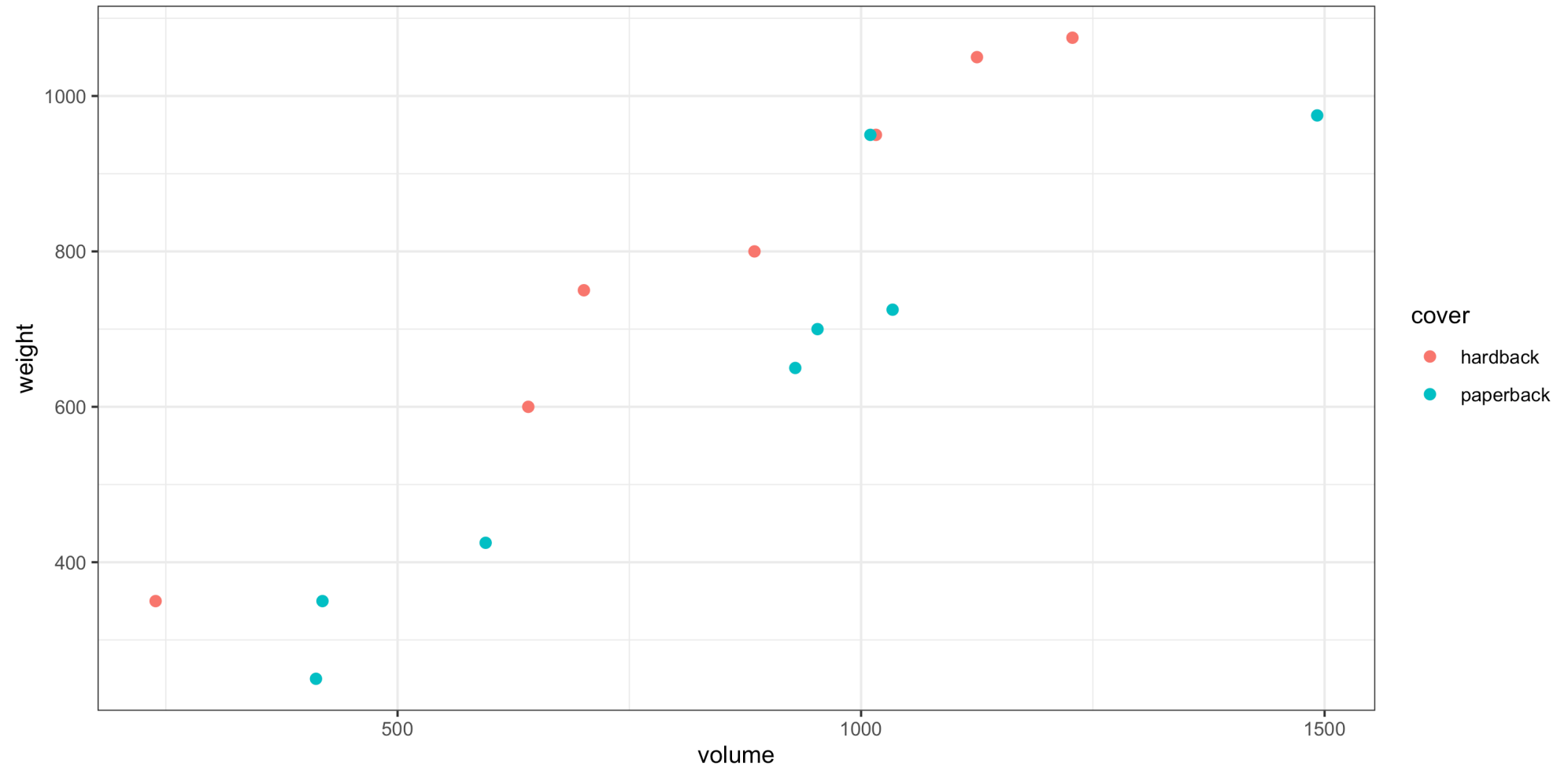
✗ scales::discard()	masks	purrr::discard()
✗ dplyr::filter()	masks	stats::filter()
✗ recipes::fixed()	masks	stringr::fixed()
✗ dplyr::lag()	masks	stats::lag()
✗ rsample::populate()	masks	Rcpp::populate()
✗ yardstick::spec()	masks	readr::spec()
✗ recipes::step()	masks	stats::step()

Book data

```
1 (books = DAAG::allbacks |>
2   as_tibble() |>
3   select(-area) |>
4   mutate(
5     cover = forcats::fct_recode(
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
11   1492     975 paperback
12    419     350 paperback
13   1010     950 paperback
14    595     425 paperback
15   1034     725 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=books)
```

parsnip model object

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

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 = boo

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,
4       data = books)
```

```
1 summary(lm_tm)
```

	Length	Class	Mode
lvl	0	-none-	NULL
ordered	1	-none-	logical
spec	7	linear_reg	list
fit	13	lm	list
preproc	1	-none-	list
elapsed	2	-none-	list
censor_probs	0	-none-	list

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

```
1 summary(lm_b)
```

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

Residuals:

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

Coefficients:

	Estimate	Std. Error	t	va
(Intercept)	161.58654	86.51918	1.	
volume	0.76159	0.09718	7.	
coverpaperback	-120.21407	115.65899	-1.	
volume:coverpaperback	-0.07573	0.12802	-0.	

Pr(>|t|)

(Intercept)	0.0887	.
volume	7.94e-06	***
coverpaperback	0.3209	
volume:coverpaperback	0.5661	

Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
1 summary(lm_tm$fit)
```

Call:

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

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	***
coverpaperback	0.3209	
volume:coverpaperback	0.5661	



Tidy coefficients

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

```
# A tibble: 4 × 5
```

	term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <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

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

```
# A tibble: 4 × 5
```

	term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <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 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

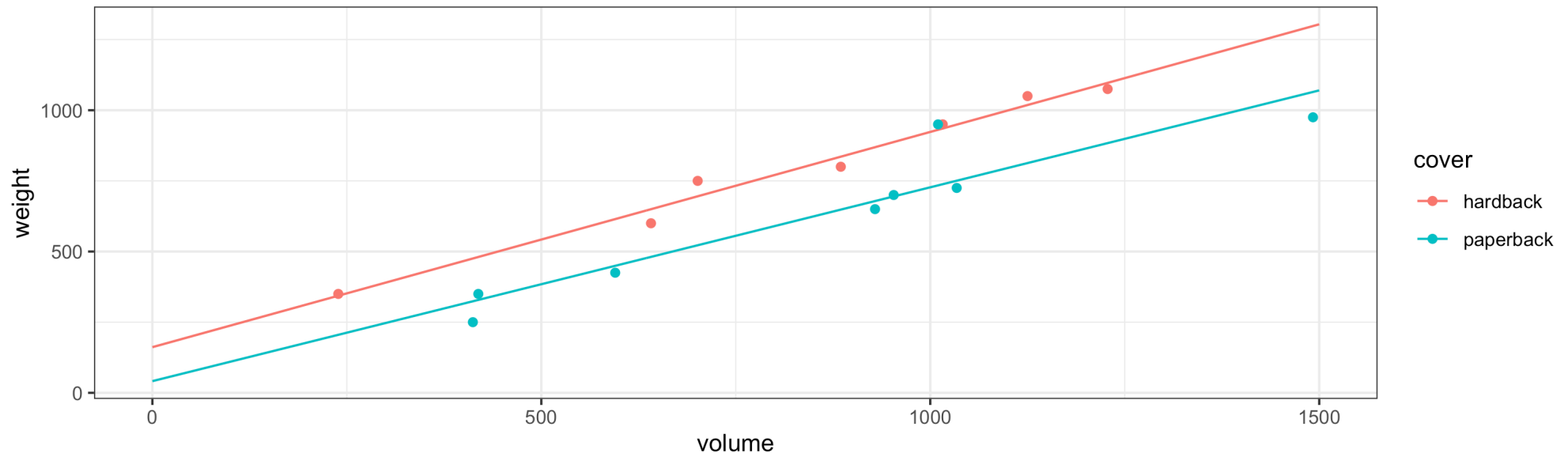
```
1 broom::augment(lm_tm, new_data = books)
```

```
# 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
11	1065.	-89.7	1492	975	paperback
12	329.	21.3	419	350	paperback
13	734.	216.	1010	950	paperback
14	449.	-24.5	595	425	paperback
15	751.	-25.6	1034	725	paperback

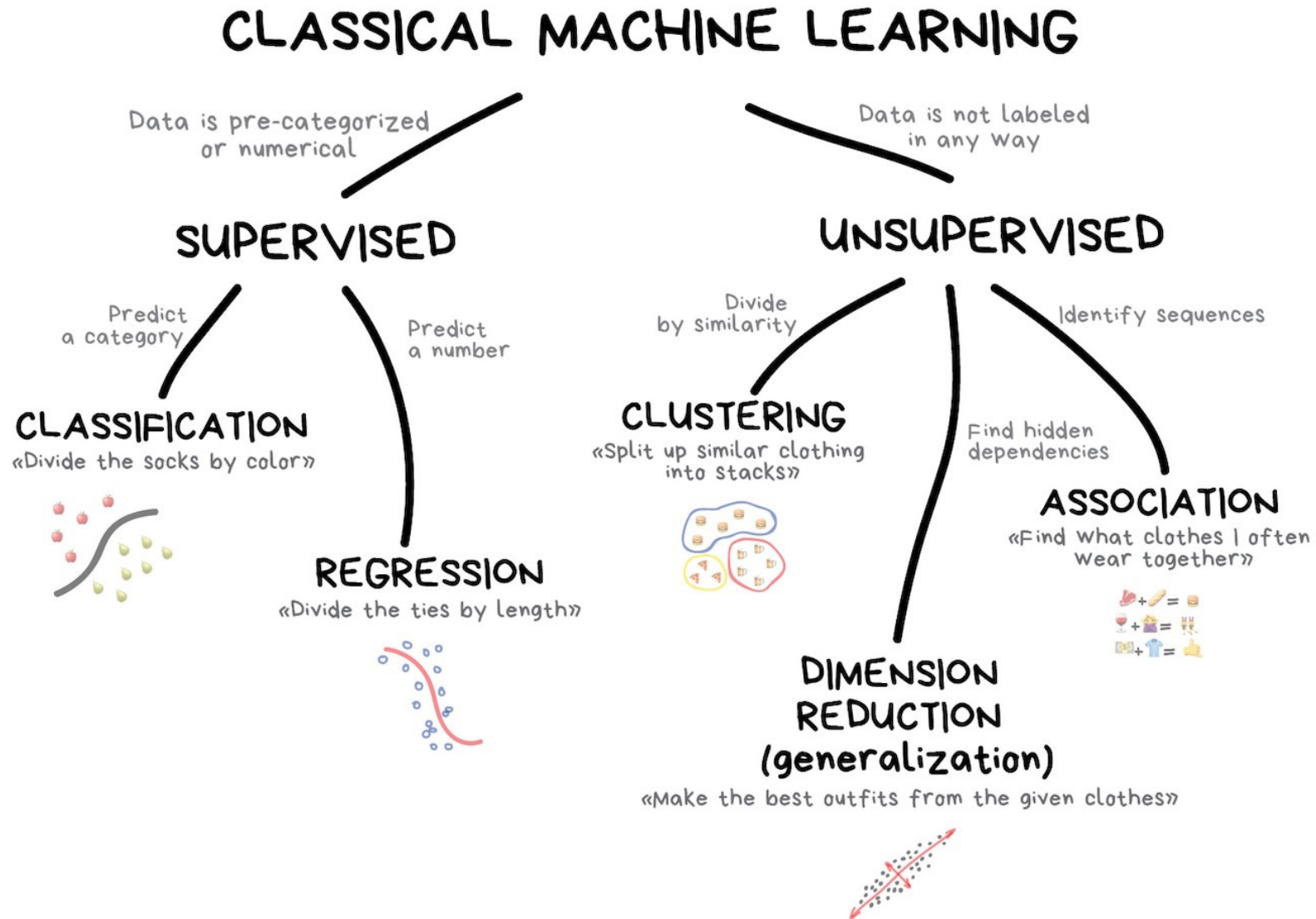
Putting it together

```
1 lm_tm |>
2   augment(
3     new_data = tidyr::expand_grid(
4       volume = seq(0, 1500, by=5),
5       cover = c("hardback", "paperback") |> as.factor()
6     )
7   ) |>
8   rename(weight = .pred) |>
9   ggplot(aes(x = volume, y = weight, color = cover, group = cover)) +
10     geom_line() +
11     geom_point(data = books)
```





Models?



Parsnip models

tidymodels.org/find/parsnip/

Other linear regression engines

```
1 show_engines("linear_reg")
```

A tibble: 8 × 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
8	quantreg	quantile 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`

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

Warning: There were 36 divergent transitions after warmup. See <https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup> to find out why this is a problem and how to eliminate them.

Warning: Examine the `pairs()` plot to diagnose sampling problems

`parsnip` model object

`stan_glm`

```
family:      gaussian [identity]
formula:      weight ~ volume * cover
observations: 15
predictors:   4
```

	Median	MAD_SD
(Intercept)	94.9	60.4
volume	0.8	0.1
coverpaperback	-0.4	3.6
volume:coverpaperback	-0.2	0.0

Auxiliary parameter(s):

	Median	MAD_SD
sigma	84.9	17.6

- * For help interpreting the printed output see `?print.stanreg`
- * For info on the priors used see `?prior_summary.stanreg`

What was actually run?

```
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 `generics::tidy()`:
```

```
! `x` seems to be outputted from the rstanarm package.
```

```
i Tidiers for mixed model output now live in broom.mixed.
```

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

```
# A tibble: 4 × 3
```

	term	estimate	std.error
	<chr>	<dbl>	<dbl>
1	(Intercept)	94.9	60.4
2	volume	0.834	0.0734
3	coverpaperback	-0.354	3.58
4	volume:coverpaperback	-0.199	0.0498

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

```
# A tibble: 1 × 4
```

	algorithm	pss	nobs	sigma
	<chr>	<dbl>	<int>	<dbl>
1	sampling	4000	15	84.9

Augment

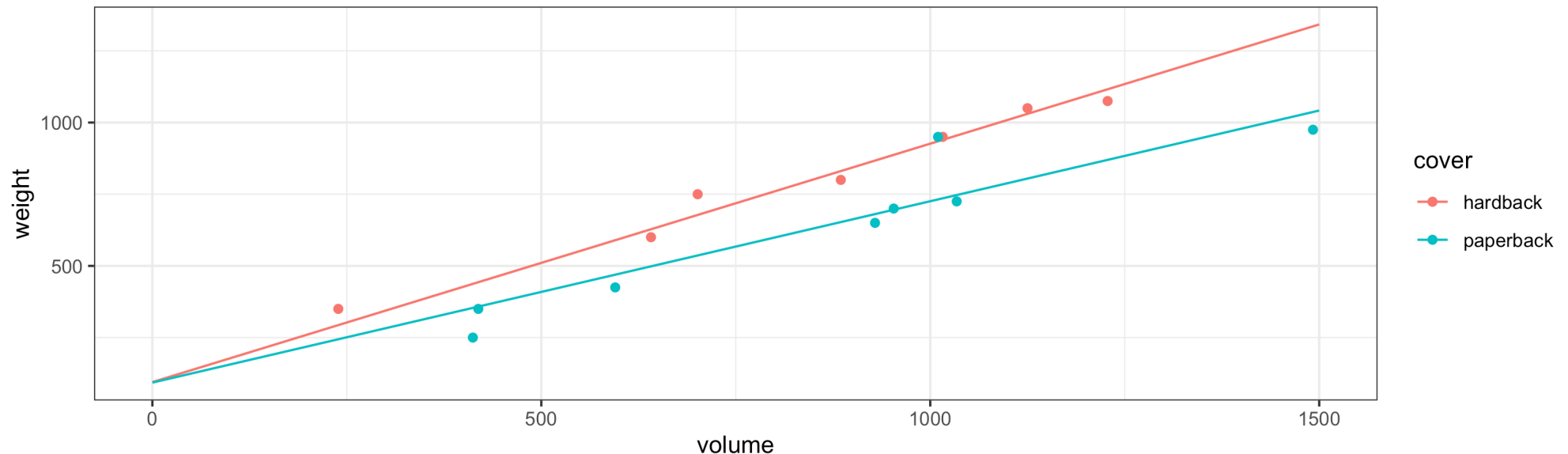
```
1 augment(bayes_tm, new_data=books)
```

```
# A tibble: 15 × 5
```

	.pred <dbl>	.resid <dbl>	volume <dbl>	weight <dbl>	cover <fct>
1	831.	-30.7	885	800	hardback
2	940.	10.3	1016	950	hardback
3	1030.	19.7	1125	1050	hardback
4	293.	56.5	239	350	hardback
5	678.	72.3	701	750	hardback
6	628.	-27.8	641	600	hardback
7	1116.	-41.0	1228	1075	hardback
8	354.	-104.	412	250	paperback
9	696.	4.18	953	700	paperback
10	681.	-30.6	929	650	paperback
11	1037.	-61.8	1492	975	paperback
12	358.	-8.00	419	350	paperback
13	732.	218.	1010	950	paperback
14	469.	-44.3	595	425	paperback
15	747.	-22.1	1034	725	paperback

Predictions

```
1 bayes_tm |>
2   augment(
3     new_data = tidyr::expand_grid(
4       volume = seq(0, 1500, by=5),
5       cover = c("hardback", "paperback") |> as.factor()
6     )
7   ) |>
8   rename(weight = .pred) |>
9   ggplot(aes(x = volume, y = weight, color = cover, group = cover)) +
10     geom_line() +
11     geom_point(data = books)
```





Performance

```
1 lm_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
```

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

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 mae     standard        46.6
```

```
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        72.1
```

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

```
# A tibble: 1 × 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 mae     standard        50.1
```

Cross validation and Feature engineering

The Office & IMDB

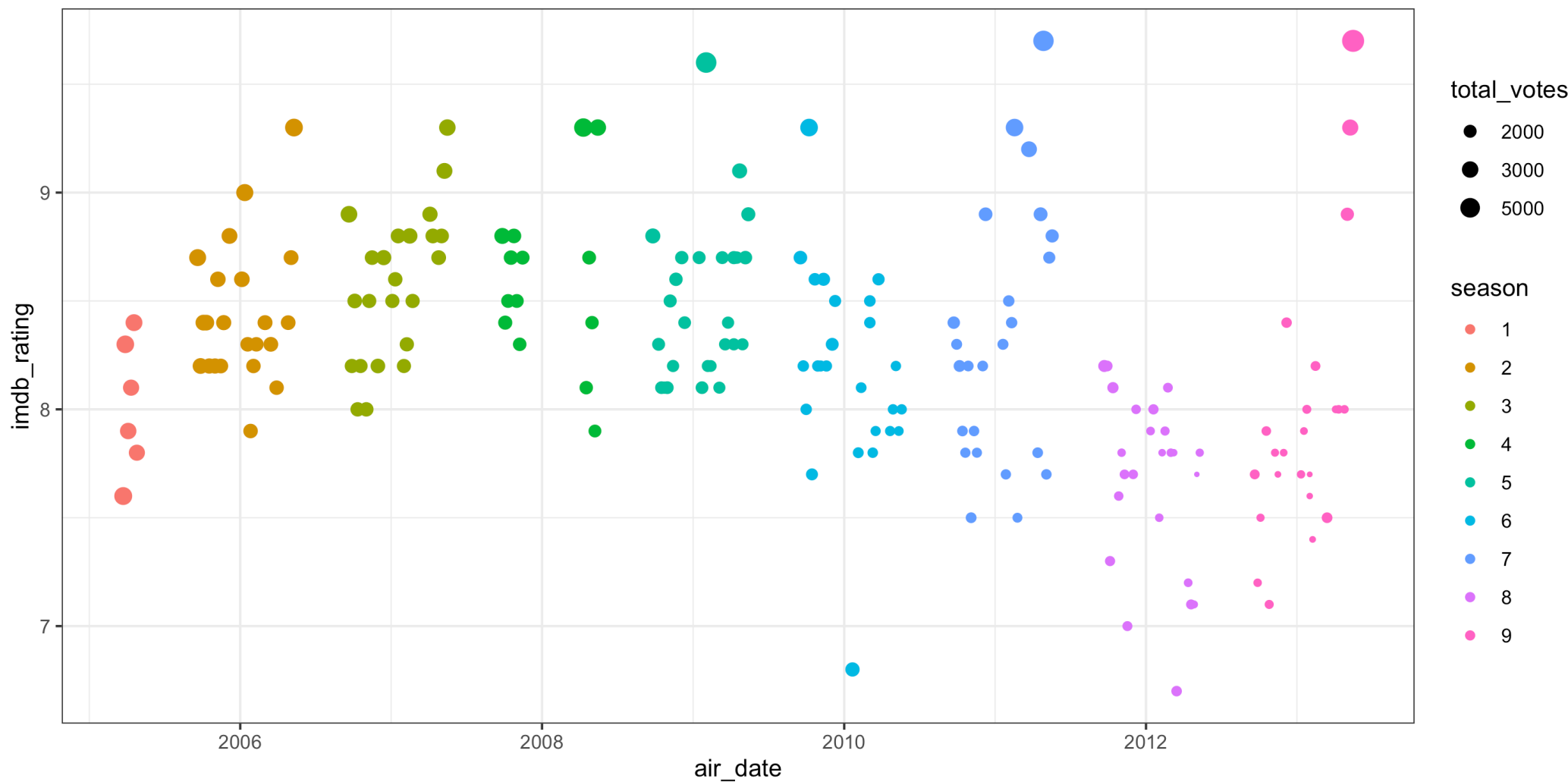
```
1 (office_ratings = read_csv("data/office_ratings.csv"))
```

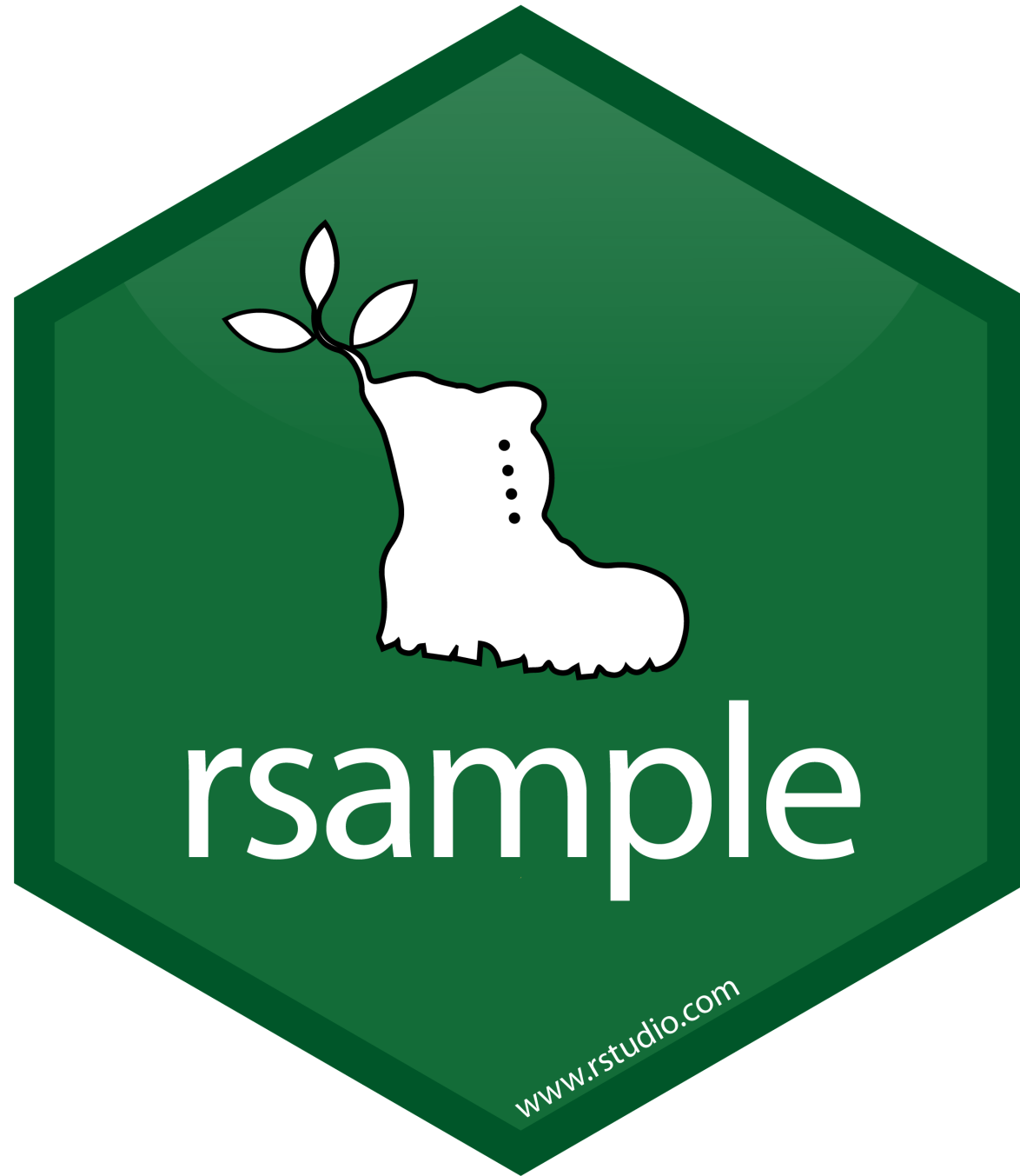
```
# A tibble: 188 × 6
```

	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_vot
	<dbl>	<dbl>	<chr>	<dbl>	<db
1	8	18	Last Da...	7.8	14
2	9	14	Vandal...	7.6	14
3	2	8	Perform...	8.2	24
4	9	5	Here Co...	7.1	15
5	3	22	Beach G...	9.1	27
6	7	1	Nepotism	8.4	18
7	3	15	Phyllis...	8.3	22
8	9	21	Livin' ...	8.9	20
9	9	18	Promos	8	14
10	8	12	Pool Pa...	8	16

```
# 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_vot
	<dbl>	<dbl>	<chr>	<dbl>	<db
1	1	2	Diversi...	8.3	35
2	2	4	The Fire	8.4	27
3	2	9	E-Mail ...	8.4	25
4	2	12	The Inj...	9	32
5	2	22	Casino ...	9.3	36
6	3	5	Initiat...	8.2	22
7	3	16	Busines...	8.8	26
8	3	17	Cocktai...	8.5	22
9	4	6	Branch ...	8.5	21
10	4	7	Survivo...	8.3	21

```
# 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

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

```
1 summary(r)
```

A tibble: 6 × 4

	variable <chr>	type <list>	role <chr>	source <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"))
```

Adding Holidays

```
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  )
```

Seasons as factors

```
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))
```

Dummy coding

```
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())
```

top10_votes

```
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)
```

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

```
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_USNewYearsE...
	<dbl>	<chr>	<dbl>	<int>	<int>	<int>
1	18	Last Da...	7.8	0	0	
2	14	Vandali...	7.6	0	0	
3	8	Perform...	8.2	0	0	
4	5	Here Co...	7.1	0	0	
5	22	Beach G...	9.1	0	0	
6	1	Nepotism	8.4	0	0	
7	15	Phyllis...	8.3	0	0	
8	21	Livin' ...	8.9	0	0	
9	18	Promos	8	0	0	
10	12	Pool Pa...	8	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>,
# 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>, ...
```

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
2	1	1
air_date_month_Aug	air_date_month_Sep	air_date_month_Oct
1	2	2
air_date_month_Nov	air_date_month_Dec	top10
2	2	2

Removing zero variance predictors

```
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())
```

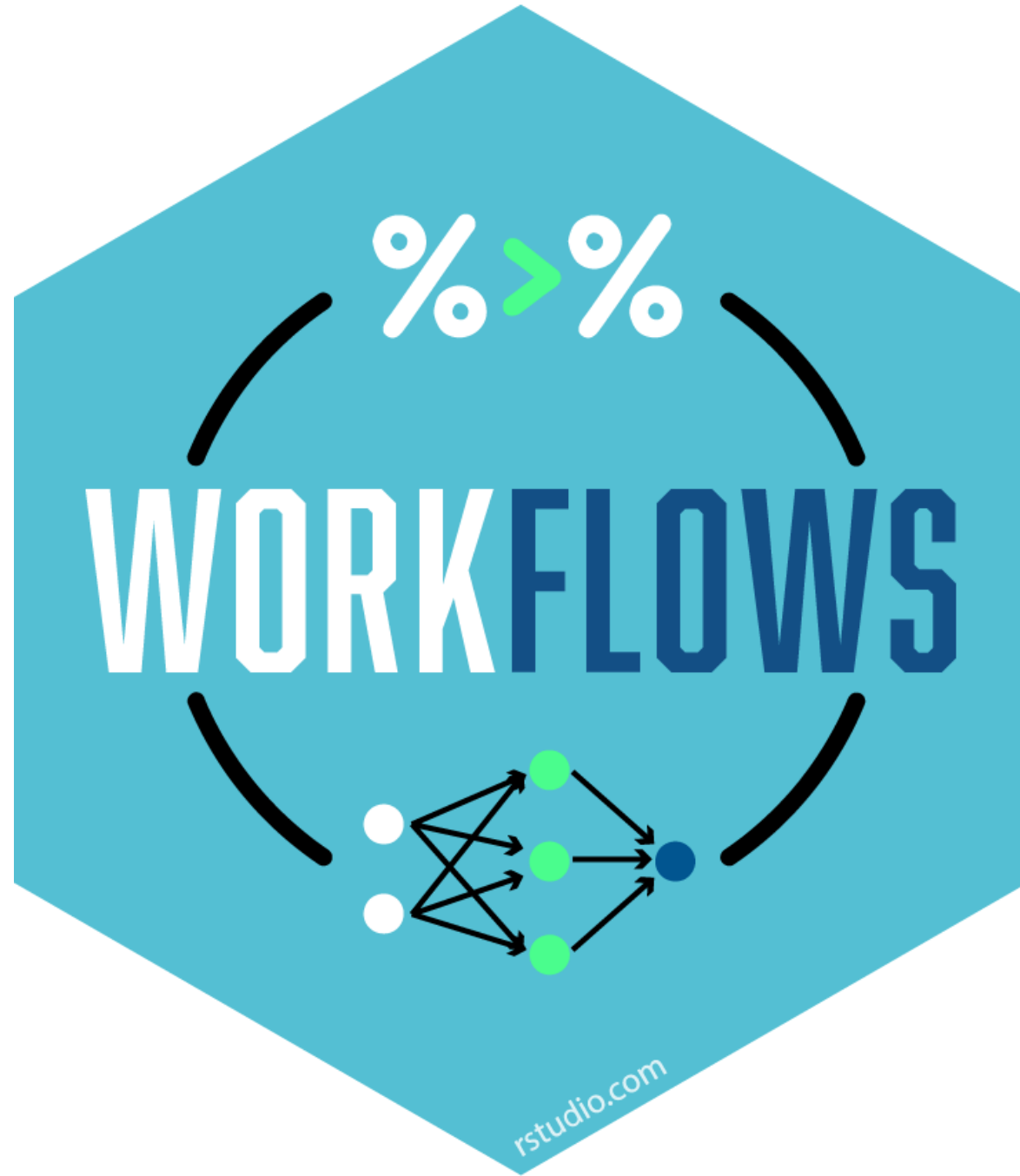
```
1 prep(r) |>
2   bake(new_data = office_train)
```

A tibble: 150 × 22

	episode	title	imdb_rating	season_X2	season_X3	season_X4	season_X5	season_X6	season_X7	season_X8
	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	18	Last D...	7.8	0	0	0	0	0	0	0
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	0

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>,
air_date_month_Nov <dbl>, air_date_month_Dec <dbl>, top10 <int>



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 —

Linear Regression Model Specification (regression)

Computational engine: lm

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 —

Call:

stats::lm(formula = ..y ~ ., data = data)

Coefficients:

(Intercept)

episode

season X2

season X3

season X4

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 )
```

→ A | warning: prediction from rank-deficient fi

There were issues with some computations A: x1

There were issues with some computations A: x1

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]>	<tibbl
2	<split [120/30]>	Fold2	<tibble [2 × 4]>	<tibbl
3	<split [120/30]>	Fold3	<tibble [2 × 4]>	<tibbl
4	<split [120/30]>	Fold4	<tibble [2 × 4]>	<tibbl
5	<split [120/30]>	Fold5	<tibble [2 × 4]>	<tibbl

There were issues with some computations:

– Warning(s) x1: prediction from rank-deficien

Run `show\_notes(.Last.tune.result)` for more inf

Fold performance

```
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	pre0_mod0_post0
2	rsq	standard	0.429	5	0.0597	pre0_mod0_post0

```
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	pre0_mod0_post0
2	Fold2	rmse	standard	0.403	pre0_mod0_post0
3	Fold3	rmse	standard	0.405	pre0_mod0_post0
4	Fold4	rmse	standard	0.454	pre0_mod0_post0
5	Fold5	rmse	standard	0.368	pre0_mod0_post0