List columns

Daniel Anderson
Week 4, Class 2

Agenda

- Finish up last few slides from Monday
- Review Lab 2
- Introduce list columns
- In-class midterm (last 20 minutes)

Learning objectives

- Understand list columns and how they relate to `base::split`
- Fluently nest/unnest data frames
- Understand why `tidyr::nest` can be a powerful framework (data frames) and when `tidyr::unnest` can/should be used to move out of nested data frames and into a standard data frame.

Review Lab 2

Setup

(please follow along)

```
library(tidyverse)
library(fs)
files <- dir_ls(here::here("data"), glob = "*.csv")
d <- files %>%
    map_df(read_csv, .id = "file") %>%
    mutate(file = str_replace_all(file, here::here("data"), ""),
           grade = str_replace_all(file,
                                   "/g(\\d?\\d).+", "\\1"),
           grade = as.integer(grade),
           year = str_replace_all(file,
                                  ".+files(\\d\\d)_sim.+",
                                  "\\1"),
           year = as.integer(year),
           content = str_replace_all(file,
                                     "/g\\d?\\d(.+)pfiles.+",
                                     "\\1")) %>%
    select(-file) %>%
    select(ssid, grade, year, content, testeventid, asmtprmrydsbltycd,
           asmtscndrydsbltycd, Entry:WMLE)
```

Comparing models

Let's say we wanted to fit/compare a set of models for each content area

1. `lm(Theta ~ asmtprmrydsbltycd)`
2. `lm(Theta ~ asmtprmrydsbltycd + asmtscndrydsbltycd)`
3. `lm(Theta ~ asmtprmrydsbltycd + asmtscndrydsbltycd + asmtprmrydsbltycd:asmtscndrydsbltycd)`

Data pre-processing

- The disability variables are stored as numbers, we need them as factors
- We'll make the names easier in the process

```
d <- d %>%
    mutate(primary = as.factor(asmtprmrydsbltycd),
           secondary = as.factor(asmtscndrydsbltycd))
```

If you're interested in what the specific codes refer to, see here.

Split the data

The base method we've been using...

```
splt_content <- split(d, d$content)
str(splt_content)
```

```
## List of 5
##  $ ELA    :Classes 'tbl_df', 'tbl' and 'data.frame':    3627 obs. of  27 variab
##   ..$ ssid                : num [1:3627] 9466908 7683685 9025693 10099824 18886
##   ..$ grade               : int [1:3627] 11 11 11 11 11 11 11 11 11 11 ...
##   ..$ year                : int [1:3627] 18 18 18 18 18 18 18 18 18 18 ...
##   ..$ content             : chr [1:3627] "ELA" "ELA" "ELA" "ELA" ...
##   ..$ testeventid         : num [1:3627] 148933 147875 143699 143962 150680 ...
##   ..$ asmtprmrydsbltycd   : num [1:3627] 0 10 40 82 10 80 50 10 50 82 ...
##   ..$ asmtscndrydsbltycd  : num [1:3627] 0 0 20 0 0 80 0 0 0 0 ...
##   ..$ Entry               : num [1:3627] 123 88 105 153 437 307 305 42 59 304 .
##   ..$ Theta               : num [1:3627] 1.27 1.55 3.28 4.48 2.67 ...
##   ..$ Status              : num [1:3627] 1 1 1 1 1 0 1 1 1 0 ...
##   ..$ Count               : num [1:3627] 36 36 36 36 36 36 36 36 36 36 ...
##   ..$ RawScore            : num [1:3627] 23 25 33 35 31 36 34 18 3 36 ...
##   ..$ SE                  : num [1:3627] 0.371 0.385 0.619 1.023 0.501 ...
##   ..$ Infit               : num [1:3627] 0.93 0.95 0.9 0.93 0.92 1 1.06 1.55 0.
##   ..$ Infit_Z             : num [1:3627] -0.34 -0.37 -0.04 0.23 -0.18 0 0.31 2.
##   ..$ Outfit              : num [1:3627] 0.82 0.81 1.63 0.35 0.88 1 0.86 1.74 0
```

We could use this method

```
m1 <- map(splt_content, ~lm(Theta ~ asmtprmrydsbltycd, data = .x))

m2 <- map(splt_content, ~lm(Theta ~ asmtprmrydsbltycd +
                                   asmtscndrydsbltycd,
                           data = .x))

m3 <- map(splt_content, ~lm(Theta ~ asmtprmrydsbltycd *
                                   asmtscndrydsbltycd,
                           data = .x))
```

- We could then go through and conduct tests to see which model was better, etc.

Alternative

- Create a data frame with a list column

```
by_content <- d %>%
    nest(-content)
by_content
```

```
## # A tibble: 5 x 2
##   content data
##   <chr>   <list>
## 1 ELA     <tibble [3,627 × 26]>
## 2 Math    <tibble [3,629 × 26]>
## 3 Rdg     <tibble [3,627 × 26]>
## 4 Science <tibble [1,435 × 26]>
## 5 Wri     <tibble [3,627 × 26]>
```

What's going on here?

```
str(by_content$data)
```

```
## List of 5
##  $ :Classes 'tbl_df', 'tbl' and 'data.frame':    3627 obs. of  26 variables:
##   ..$ ssid                : num [1:3627] 9466908 7683685 9025693 10099824 18886
##   ..$ grade               : int [1:3627] 11 11 11 11 11 11 11 11 11 11 ...
##   ..$ year                : int [1:3627] 18 18 18 18 18 18 18 18 18 18 ...
##   ..$ testeventid         : num [1:3627] 148933 147875 143699 143962 150680 ...
##   ..$ asmtprmrydsbltycd   : num [1:3627] 0 10 40 82 10 80 50 10 50 82 ...
##   ..$ asmtscndrydsbltycd  : num [1:3627] 0 0 20 0 0 80 0 0 0 0 ...
##   ..$ Entry               : num [1:3627] 123 88 105 153 437 307 305 42 59 304 .
##   ..$ Theta               : num [1:3627] 1.27 1.55 3.28 4.48 2.67 ...
##   ..$ Status              : num [1:3627] 1 1 1 1 1 0 1 1 1 0 ...
##   ..$ Count               : num [1:3627] 36 36 36 36 36 36 36 36 36 36 ...
##   ..$ RawScore            : num [1:3627] 23 25 33 35 31 36 34 18 3 36 ...
##   ..$ SE                  : num [1:3627] 0.371 0.385 0.619 1.023 0.501 ...
##   ..$ Infit               : num [1:3627] 0.93 0.95 0.9 0.93 0.92 1 1.06 1.55 0.
##   ..$ Infit_Z             : num [1:3627] -0.34 -0.37 -0.04 0.23 -0.18 0 0.31 2.
##   ..$ Outfit              : num [1:3627] 0.82 0.81 1.63 0.35 0.88 1 0.86 1.74 0
##   ..$ Outfit_Z            : num [1:3627] -0.62 -0.56 1.03 -0.16 -0.12 0 0.17 3.
##   ..$ Displacement        : num [1:3627] 0.0018 0.0019 0.0022 0.0023 0.0021 0.0
##   ..$ PointMeasureCorr    : num [1:3627] 0.42 0.42 0.3 0.27 0.31 0 0.14 -0.12 0
##   ..$ Weight              : num [1:3627] 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ ObservMatch         : num [1:3627] 75 80.6 91.7 97.2 86.1 100 94.4 50 97.
```

Explore a bit

```
map_dbl(by_content$data, nrow)
```

```
## [1] 3627 3629 3627 1435 3627
```

```
map_dbl(by_content$data, ncol)
```

```
## [1] 26 26 26 26 26
```

```
map_dbl(by_content$data, ~mean(.x$Theta))
```

```
## [1]  1.28001056 -0.06683086  1.37068376  1.57850321  1.26090709
```

It's a data frame!

We can add these summaries if we want

```
by_content %>%
    mutate(n = map_dbl(data, nrow))
```

```
## # A tibble: 5 x 3
##   content data                    n
##   <chr>   <list>                <dbl>
## 1 ELA     <tibble [3,627 × 26]>  3627
## 2 Math    <tibble [3,629 × 26]>  3629
## 3 Rdg     <tibble [3,627 × 26]>  3627
## 4 Science <tibble [1,435 × 26]>  1435
## 5 Wri     <tibble [3,627 × 26]>  3627
```

`map_*`

- Note on the previous example we used `map_dbl` and we got a vector in return.
- What would happen if we just used `map`?

map_*

- Note on the previous example we used `map_dbl` and we got a vector in return.
- What would happen if we just used `map`?

```
by_content %>%
    mutate(n = map(data, nrow))
```

```
## # A tibble: 5 x 3
##   content data                    n
##   <chr>   <list>                  <list>
## 1 ELA     <tibble [3,627 × 26]> <int [1]>
## 2 Math    <tibble [3,629 × 26]> <int [1]>
## 3 Rdg     <tibble [3,627 × 26]> <int [1]>
## 4 Science <tibble [1,435 × 26]> <int [1]>
## 5 Wri     <tibble [3,627 × 26]> <int [1]>
```

Let's fit a model!

```
by_content %>%
    mutate(m1 = map(data, ~lm(Theta ~ primary, data = .x)))
```

```
## # A tibble: 5 x 3
##   content data                    m1
##   <chr>   <list>                  <list>
## 1 ELA     <tibble [3,627 × 26]> <S3: lm>
## 2 Math    <tibble [3,629 × 26]> <S3: lm>
## 3 Rdg     <tibble [3,627 × 26]> <S3: lm>
## 4 Science <tibble [1,435 × 26]> <S3: lm>
## 5 Wri     <tibble [3,627 × 26]> <S3: lm>
```

Extract the coefficients

```
by_content %>%
    mutate(m1 = map(data, ~lm(Theta ~ primary, data = .x)),
           coefs = map(m1, coef))
```

```
## # A tibble: 5 x 4
##   content data                    m1       coefs
##   <chr>   <list>                  <list>   <list>
## 1 ELA     <tibble [3,627 × 26]> <S3: lm> <dbl [11]>
## 2 Math    <tibble [3,629 × 26]> <S3: lm> <dbl [12]>
## 3 Rdg     <tibble [3,627 × 26]> <S3: lm> <dbl [11]>
## 4 Science <tibble [1,435 × 26]> <S3: lm> <dbl [12]>
## 5 Wri     <tibble [3,627 × 26]> <S3: lm> <dbl [12]>
```

Challenge

- Continue with the above, but output a data frame with three columns: `content`, `intercept`, and `TBI` (which is code 74).
- In other words, output the mean score for students who were coded as not having a disability (code 0), along with students coded as having TBI.

```r
by_content %>%
    mutate(m1 = map(data, ~lm(Theta ~ primary, data = .x)),
           coefs = map(m1, coef),
           no_disab = map_dbl(coefs, 1),
           tbi = no_disab + map_dbl(coefs, "primary74")) %>%
    select(content, no_disab, tbi)
```

```
## # A tibble: 5 x 3
##   content   no_disab       tbi
##   <chr>        <dbl>     <dbl>
## 1 ELA      0.9322336 0.1674462
## 2 Math    -0.1587907 0.1910821
## 3 Rdg      1.363101  1.629048
## 4 Science  1.491319  2.790971
## 5 Wri      1.571441  1.167429
```

Compare models

- Back to our original task - fit all three models

You try first

1. `lm(Theta ~ primary)`
2. `lm(Theta ~ primary + secondary)`
3. `lm(Theta ~ primary + secondary + primary:secondary)`

Model fits

```
mods <- by_content %>%
    mutate(m1 = map(data, ~lm(Theta ~ primary, data = .x)),
           m2 = map(data, ~lm(Theta ~ primary + secondary, data = .x)),
           m3 = map(data, ~lm(Theta ~ primary * secondary, data = .x)))
mods
```

```
## # A tibble: 5 x 5
##   content data                    m1       m2       m3
##   <chr>   <list>                  <list>   <list>   <list>
## 1 ELA     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm>
## 2 Math    <tibble [3,629 × 26]> <S3: lm> <S3: lm> <S3: lm>
## 3 Rdg     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm>
## 4 Science <tibble [1,435 × 26]> <S3: lm> <S3: lm> <S3: lm>
## 5 Wri     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm>
```

Brief foray into parallel iterations

The `stats::anova` function can compare the fit of two models

Brief foray into parallel iterations

The `stats::anova` function can compare the fit of two models

Pop Quiz

How would we extract just ELA model 1 and 2?

Brief foray into parallel iterations

The `stats::anova` function can compare the fit of two models

Pop Quiz

How would we extract just ELA model 1 and 2?

```
mods$m1[[1]]
```

```
##
## Call:
## lm(formula = Theta ~ primary, data = .x4)
##
## Coefficients:
## (Intercept)    primary10    primary20
##     0.93223      0.38570     -0.03168
##   primary60    primary70    primary74
##     0.77625     -1.83026     -0.76479
##   primary90
##     1.47936
```

```
mods$m2[[1]]
```

```
##
## Call:
## lm(formula = Theta ~ primary + secondar
##
## Coefficients:
## (Intercept)    primary10    primary20
##     0.93223      0.38570     -0.03168
##   primary60    primary70    primary74
##     0.77625     -1.83026     -0.76479
##   primary90  secondary10  secondary20
##      1.4101      -0.5685      -1.0558
## secondary50  secondary60  secondary70
##      0.2484       0.4831      -1.3336
```

Which fits better?

```
compare <- anova(mods$m1[[1]], mods$m2[[1]])
compare
```

```
## Analysis of Variance Table
##
## Model 1: Theta ~ primary
## Model 2: Theta ~ primary + secondary
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1   3616 20905
## 2   3605 20100 11    804.26 13.113 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

`map2`

- Works the same as `map` but iterates over two vectors concurrently
- Let's compare model 1 and 2

map2

- Works the same as `map` but iterates over two vectors concurrently
- Let's compare model 1 and 2

```
mods %>%
    mutate(comp12 = map2(m1, m2, anova))
```

```
## # A tibble: 5 x 6
##   content data                    m1       m2       m3       comp12
##   <chr>   <list>                  <list>   <list>   <list>   <list>
## 1 ELA     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 2 Math    <tibble [3,629 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 3 Rdg     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 4 Science <tibble [1,435 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 5 Wri     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
```

map2

- Works the same as `map` but iterates over two vectors concurrently
- Let's compare model 1 and 2

```
mods %>%
    mutate(comp12 = map2(m1, m2, anova))
```

```
## # A tibble: 5 x 6
##   content data                    m1       m2       m3       comp12
##   <chr>   <list>                  <list>   <list>   <list>   <list>
## 1 ELA     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 2 Math    <tibble [3,629 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 3 Rdg     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 4 Science <tibble [1,435 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
## 5 Wri     <tibble [3,627 × 26]> <S3: lm> <S3: lm> <S3: lm> <anova [2 × 6]>
```

Perhaps not terrifically helpful

Back to our anova object

- Can we pull out useful things?

```
str(compare)
```

```
## Classes 'anova' and 'data.frame':    2 obs. of  6 variables:
##  $ Res.Df   : num  3616 3605
##  $ RSS      : num  20905 20100
##  $ Df       : num  NA 11
##  $ Sum of Sq: num  NA 804
##  $ F        : num  NA 13.1
##  $ Pr(>F)   : num  NA 7.66e-25
##  - attr(*, "heading")= chr  "Analysis of Variance Table\n" "Model 1: Theta ~ pr
```

Back to our anova object

- Can we pull out useful things?

```
str(compare)
```

```
## Classes 'anova' and 'data.frame':    2 obs. of  6 variables:
##  $ Res.Df   : num  3616 3605
##  $ RSS      : num  20905 20100
##  $ Df       : num  NA 11
##  $ Sum of Sq: num  NA 804
##  $ F        : num  NA 13.1
##  $ Pr(>F)   : num  NA 7.66e-25
##  - attr(*, "heading")= chr  "Analysis of Variance Table\n" "Model 1: Theta ~ pr
```

Try pulling out the p value

Extract p value

- *Note - I'd recommend looking at more than just a p-value, but I do think this is useful for a quick glance*

```
compare$`Pr(>F)`
```

```
## [1]           NA 7.663566e-25
```

```
compare[["Pr(>F)"]]
```

```
## [1]           NA 7.663566e-25
```

Extract p value

- *Note - I'd recommend looking at more than just a p-value, but I do think this is useful for a quick glance*

```
compare$`Pr(>F)`
```

```
## [1]           NA 7.663566e-25
```

```
compare[["Pr(>F)"]]
```

```
## [1]           NA 7.663566e-25
```

```
compare$`Pr(>F)`[2]
```

```
## [1] 7.663566e-25
```

```
compare[["Pr(>F)"]][2]
```

```
## [1] 7.663566e-25
```

All p-values

Note - this is probably the most compact syntax, but that doesn't mean it's the most clear

```
mods %>%
    mutate(comp12 = map2(m1, m2, anova),
           p12 = map_dbl(comp12, list("Pr(>F)", 2)))
```

```
## # A tibble: 5 x 7
##   content data              m1      m2      m3      comp12               p12
##   <chr>   <list>            <list>  <list>  <list>  <list>             <dbl>
## 1 ELA     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 7.663566e-25
## 2 Math    <tibble [3,629 … <S3: l… <S3: l… <S3: l… <anova [2 … 1.724262e-22
## 3 Rdg     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 1.527172e-28
## 4 Science <tibble [1,435 … <S3: l… <S3: l… <S3: l… <anova [2 … 4.685885e-18
## 5 Wri     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 5.785623e-11
```

Slight alternative

- Write a function that pulls the p-value from model comparison objects

```
extract_p <- function(anova_ob) {
    anova_ob[["Pr(>F)"]][2]
}
```

Slight alternative

- Write a function that pulls the p-value from model comparison objects

```
extract_p <- function(anova_ob) {
    anova_ob[["Pr(>F)"]][2]
}
```

- Loop this function through the anova objects

```
mods %>%
    mutate(comp12 = map2(m1, m2, anova),
           p12 = map_dbl(comp12, extract_p))
```

```
## # A tibble: 5 x 7
##   content data              m1      m2      m3      comp12                p12
##   <chr>   <list>            <list>  <list>  <list>  <list>              <dbl>
## 1 ELA     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 7.663566e-25
## 2 Math    <tibble [3,629 … <S3: l… <S3: l… <S3: l… <anova [2 … 1.724262e-22
## 3 Rdg     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 1.527172e-28
## 4 Science <tibble [1,435 … <S3: l… <S3: l… <S3: l… <anova [2 … 4.685885e-18
## 5 Wri     <tibble [3,627 … <S3: l… <S3: l… <S3: l… <anova [2 … 5.785623e-11
```

Look at all R^2

It's a normal data frame!

```
mods %>%
    gather(model, output, m1:m3)
```

```
## # A tibble: 15 x 4
##    content data                    model output
##    <chr>   <list>                  <chr> <list>
##  1 ELA     <tibble [3,627 × 26]> m1    <S3: lm>
##  2 Math    <tibble [3,629 × 26]> m1    <S3: lm>
##  3 Rdg     <tibble [3,627 × 26]> m1    <S3: lm>
##  4 Science <tibble [1,435 × 26]> m1    <S3: lm>
##  5 Wri     <tibble [3,627 × 26]> m1    <S3: lm>
##  6 ELA     <tibble [3,627 × 26]> m2    <S3: lm>
##  7 Math    <tibble [3,629 × 26]> m2    <S3: lm>
##  8 Rdg     <tibble [3,627 × 26]> m2    <S3: lm>
##  9 Science <tibble [1,435 × 26]> m2    <S3: lm>
## 10 Wri     <tibble [3,627 × 26]> m2    <S3: lm>
## 11 ELA     <tibble [3,627 × 26]> m3    <S3: lm>
## 12 Math    <tibble [3,629 × 26]> m3    <S3: lm>
## 13 Rdg     <tibble [3,627 × 26]> m3    <S3: lm>
## 14 Science <tibble [1,435 × 26]> m3    <S3: lm>
## 15 Wri     <tibble [3,627 × 26]> m3    <S3: lm>
```

Extract all R^2

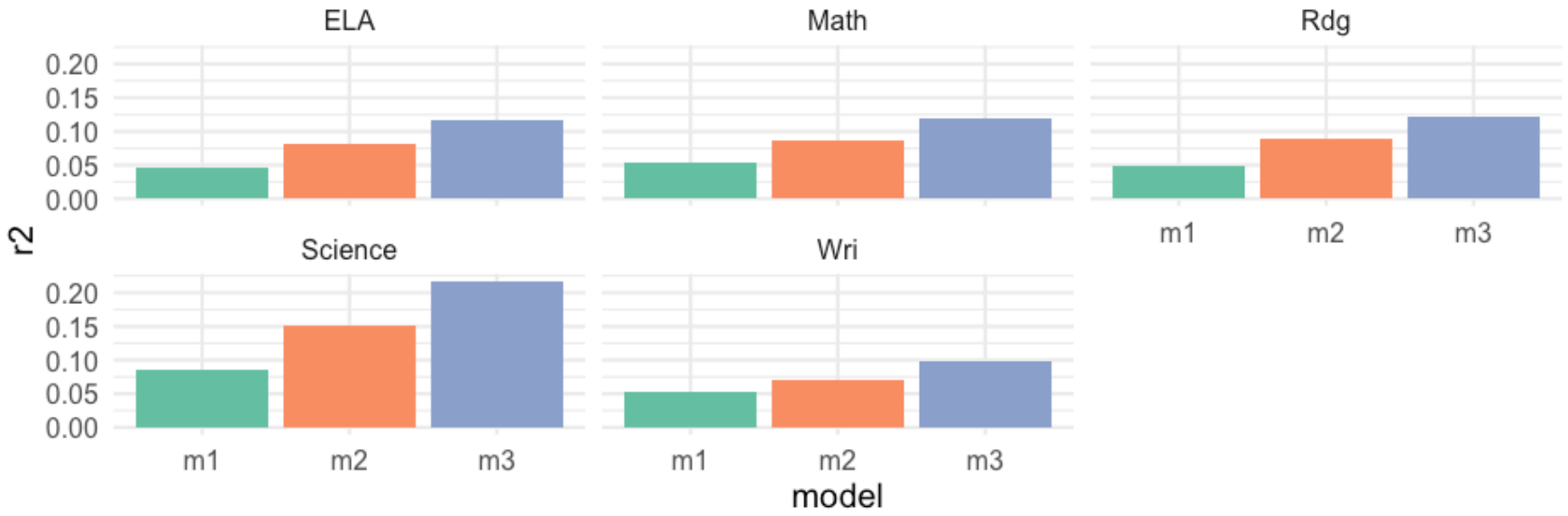
Note - might want to write a function here again

```
mods %>%
    gather(model, output, m1:m3) %>%
    mutate(r2 = map_dbl(output, ~summary(.x)$r.squared))
```

```
## # A tibble: 15 x 5
##    content data                    model output           r2
##    <chr>   <list>                  <chr> <list>        <dbl>
##  1 ELA     <tibble [3,627 × 26]> m1    <S3: lm> 0.04517421
##  2 Math    <tibble [3,629 × 26]> m1    <S3: lm> 0.05326550
##  3 Rdg     <tibble [3,627 × 26]> m1    <S3: lm> 0.04805713
##  4 Science <tibble [1,435 × 26]> m1    <S3: lm> 0.08683581
##  5 Wri     <tibble [3,627 × 26]> m1    <S3: lm> 0.05171555
##  6 ELA     <tibble [3,627 × 26]> m2    <S3: lm> 0.08190917
##  7 Math    <tibble [3,629 × 26]> m2    <S3: lm> 0.08675264
##  8 Rdg     <tibble [3,627 × 26]> m2    <S3: lm> 0.08926212
##  9 Science <tibble [1,435 × 26]> m2    <S3: lm> 0.1522437
## 10 Wri     <tibble [3,627 × 26]> m2    <S3: lm> 0.06977688
## 11 ELA     <tibble [3,627 × 26]> m3    <S3: lm> 0.1161187
## 12 Math    <tibble [3,629 × 26]> m3    <S3: lm> 0.1185931
## 13 Rdg     <tibble [3,627 × 26]> m3    <S3: lm> 0.1217497
## 14 Science <tibble [1,435 × 26]> m3    <S3: lm> 0.2170660
```

Plot

```
mods %>%
    gather(model, output, m1:m3) %>%
    mutate(r2 = map_dbl(output, ~summary(.x)$r.squared)) %>%
ggplot(aes(model, r2)) +
    geom_col(aes(fill = model)) +
    facet_wrap(~content) +
    guides(fill = "none") +
    scale_fill_brewer(palette = "Set2")
```



Unnesting

- Sometimes you just want to `unnest`

Unnesting

- Sometimes you just want to `unnest`
- Imagine we want to plot the coefficients by model... how?

Unnesting

- Sometimes you just want to `unnest`
- Imagine we want to plot the coefficients by model... how?
- `broom::tidy()` => `tidyr::unnest()`

Tidy

```
mods %>%
    gather(model, output, m1:m3) %>%
    mutate(tidied = map(output, broom::tidy))
```

```
## # A tibble: 15 x 5
##    content data                    model output   tidied
##    <chr>   <list>                  <chr> <list>   <list>
##  1 ELA     <tibble [3,627 × 26]> m1    <S3: lm> <tibble [11 × 5]>
##  2 Math    <tibble [3,629 × 26]> m1    <S3: lm> <tibble [12 × 5]>
##  3 Rdg     <tibble [3,627 × 26]> m1    <S3: lm> <tibble [11 × 5]>
##  4 Science <tibble [1,435 × 26]> m1    <S3: lm> <tibble [12 × 5]>
##  5 Wri     <tibble [3,627 × 26]> m1    <S3: lm> <tibble [12 × 5]>
##  6 ELA     <tibble [3,627 × 26]> m2    <S3: lm> <tibble [22 × 5]>
##  7 Math    <tibble [3,629 × 26]> m2    <S3: lm> <tibble [23 × 5]>
##  8 Rdg     <tibble [3,627 × 26]> m2    <S3: lm> <tibble [22 × 5]>
##  9 Science <tibble [1,435 × 26]> m2    <S3: lm> <tibble [22 × 5]>
## 10 Wri     <tibble [3,627 × 26]> m2    <S3: lm> <tibble [22 × 5]>
## 11 ELA     <tibble [3,627 × 26]> m3    <S3: lm> <tibble [95 × 5]>
## 12 Math    <tibble [3,629 × 26]> m3    <S3: lm> <tibble [88 × 5]>
## 13 Rdg     <tibble [3,627 × 26]> m3    <S3: lm> <tibble [91 × 5]>
## 14 Science <tibble [1,435 × 26]> m3    <S3: lm> <tibble [77 × 5]>
## 15 Wri     <tibble [3,627 × 26]> m3    <S3: lm> <tibble [94 × 5]>
```

Select and unnest

```
tidied <- mods %>%
    gather(model, output, m1:m3) %>%
    mutate(tidied = map(output, broom::tidy)) %>%
    select(content, model, tidied) %>%
    unnest()
tidied
```

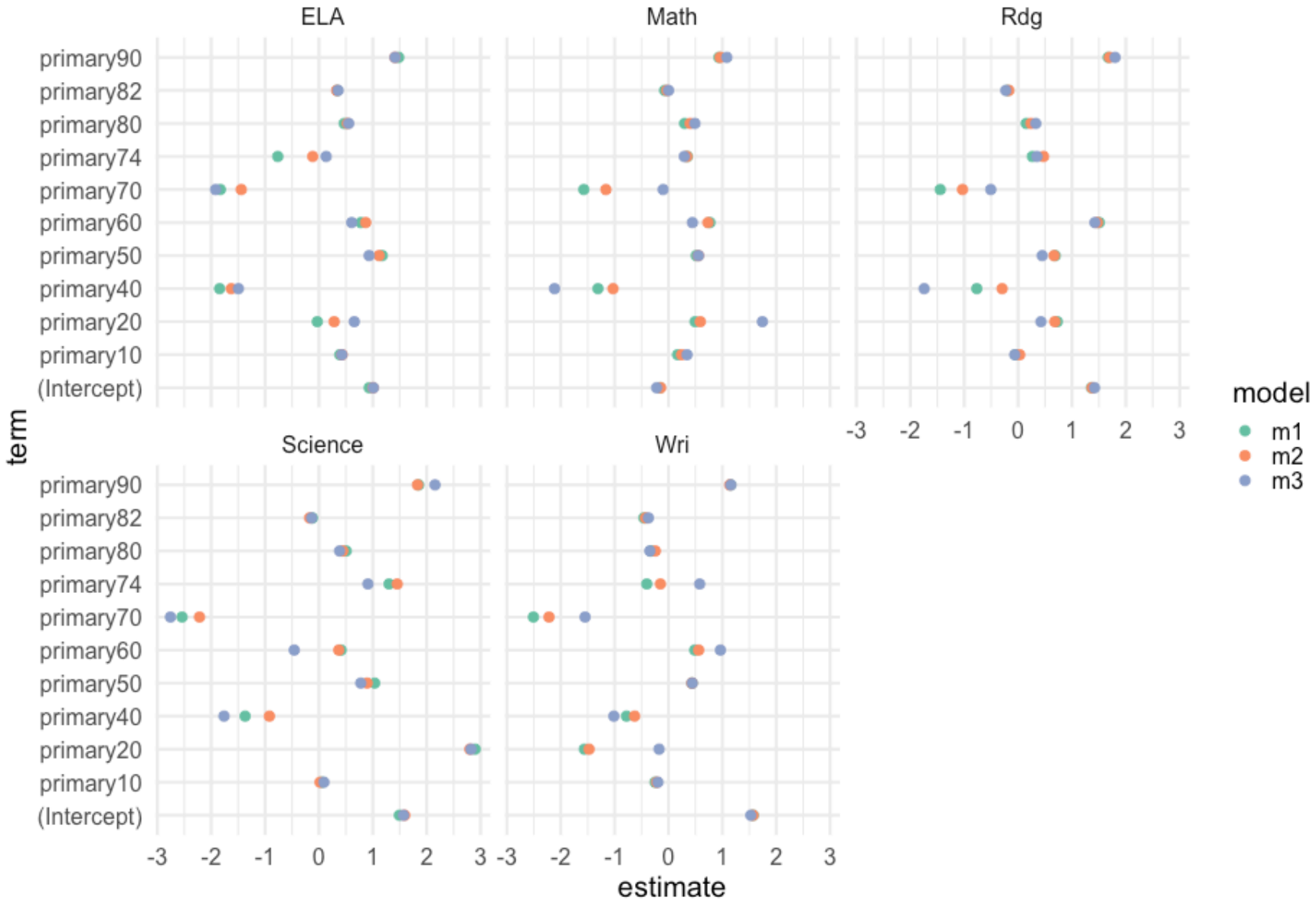
```
## # A tibble: 614 x 7
##    content model term           estimate std.error   statistic     p.value
##    <chr>   <chr> <chr>             <dbl>     <dbl>       <dbl>       <dbl>
##  1 ELA     m1    (Intercept)  0.9322336  0.2150561  4.334839   1.498396e-5
##  2 ELA     m1    primary10    0.3856986  0.2242965  1.719593   8.559207e-2
##  3 ELA     m1    primary20   -0.03167527 0.7266436 -0.04359120 9.652327e-1
##  4 ELA     m1    primary40   -1.844343   0.5559031 -3.317741   9.164595e-4
##  5 ELA     m1    primary50    1.173722   0.2890447  4.060694   4.996391e-5
##  6 ELA     m1    primary60    0.7762539  0.3866313  2.007737   4.474555e-2
##  7 ELA     m1    primary70   -1.830257   0.3086128 -5.930595   3.301860e-9
##  8 ELA     m1    primary74   -0.7647874  0.5182670 -1.475663   1.401215e-1
##  9 ELA     m1    primary80    0.4676481  0.2428640  1.925556   5.423822e-2
## 10 ELA     m1    primary82    0.3382547  0.2267600  1.491686   1.358687e-1
## # … with 604 more rows
```

Plot

Lets look how the primary coefficients change

```
to_plot <- names(coef(mods$m1[[1]]))

tidied %>%
    filter(term %in% to_plot) %>%
ggplot(aes(estimate, term, color = model)) +
    geom_point() +
    scale_color_brewer(palette = "Set2") +
    facet_wrap(~content)
```


ELA
Math
Rdg
Science
Wri
primary90
primary82
primary80
primary74
primary70
primary60
primary50
primary40
primary20
primary10
(Intercept)
term
-3
-2
-1
0
1
2
3
estimate
model
m1
m2
m3

Last bit

- We've kind of been running the wrong models this whole time
- We forgot about grade!
- No problem, just change the grouping factor

By grade

```
by_grade_content <- d %>%
    nest(-content, -grade)
by_grade_content
```

```
## # A tibble: 31 x 3
##    grade content data
##    <int> <chr>   <list>
##  1    11 ELA     <tibble [453 × 25]>
##  2    11 Math    <tibble [460 × 25]>
##  3    11 Rdg     <tibble [453 × 25]>
##  4    11 Science <tibble [438 × 25]>
##  5    11 Wri     <tibble [453 × 25]>
##  6     3 ELA     <tibble [540 × 25]>
##  7     3 Math    <tibble [536 × 25]>
##  8     3 Rdg     <tibble [540 × 25]>
##  9     3 Wri     <tibble [540 × 25]>
## 10     4 ELA     <tibble [585 × 25]>
## # … with 21 more rows
```

Fit models

```
mods_grade <- by_grade_content %>%
    mutate(m1 = map(data, ~lm(Theta ~ primary, data = .x)),
           m2 = map(data, ~lm(Theta ~ primary + secondary, data = .x)),
           m3 = map(data, ~lm(Theta ~ primary * secondary, data = .x)))
mods_grade
```

```
## # A tibble: 31 x 6
##    grade content data                m1       m2       m3
##    <int> <chr>   <list>              <list>   <list>   <list>
##  1    11 ELA     <tibble [453 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  2    11 Math    <tibble [460 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  3    11 Rdg     <tibble [453 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  4    11 Science <tibble [438 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  5    11 Wri     <tibble [453 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  6     3 ELA     <tibble [540 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  7     3 Math    <tibble [536 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  8     3 Rdg     <tibble [540 × 25]> <S3: lm> <S3: lm> <S3: lm>
##  9     3 Wri     <tibble [540 × 25]> <S3: lm> <S3: lm> <S3: lm>
## 10     4 ELA     <tibble [585 × 25]> <S3: lm> <S3: lm> <S3: lm>
## # … with 21 more rows
```

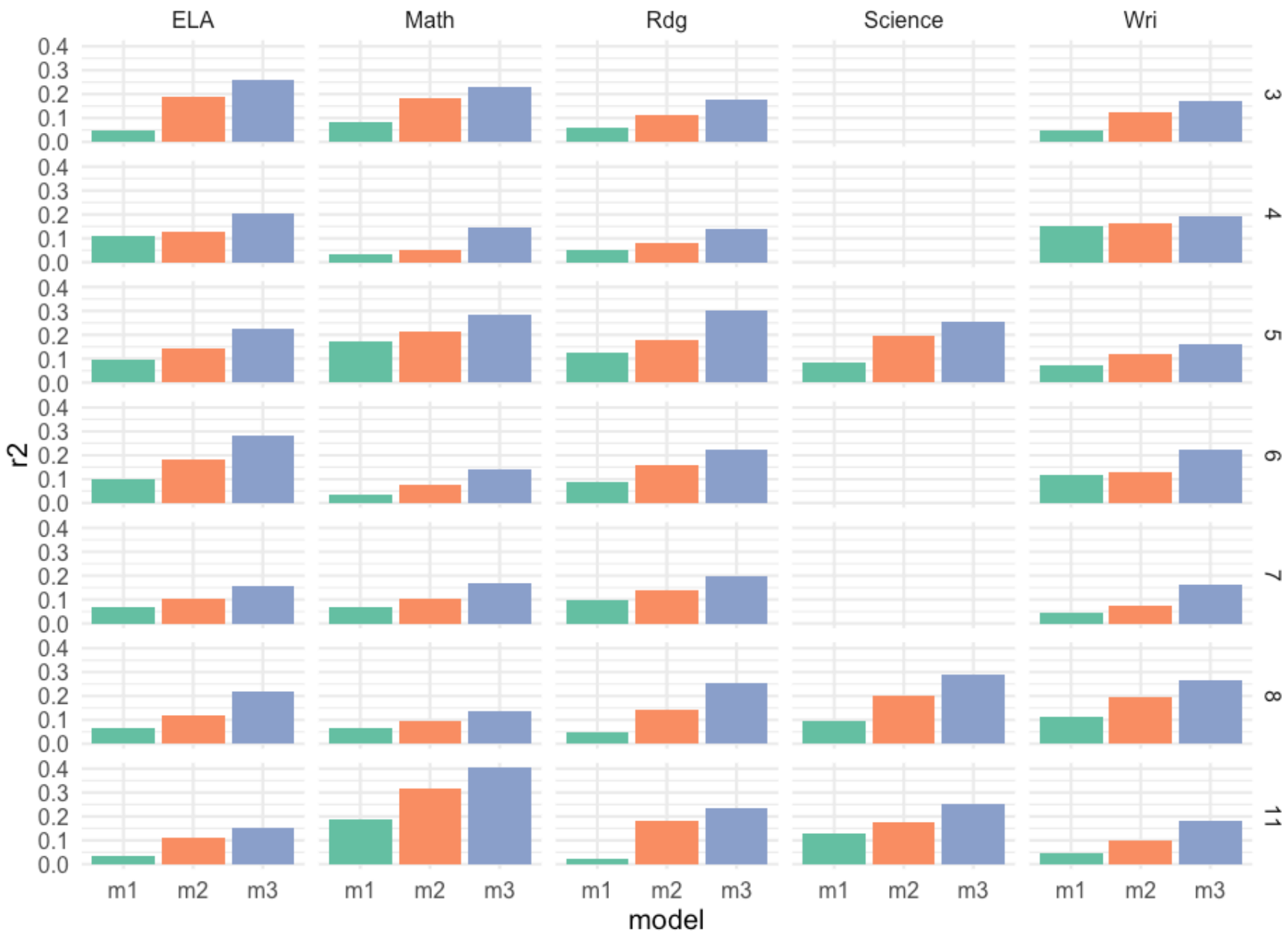
Look at R^2

```
mods_grade %>%
    gather(model, output, m1:m3) %>%
    mutate(r2 = map_dbl(output, ~summary(.x)$r.squared))
```

```
## # A tibble: 93 x 6
##    grade content data                model output           r2
##    <int> <chr>   <list>              <chr> <list>        <dbl>
##  1    11 ELA     <tibble [453 × 25]> m1    <S3: lm> 0.03353818
##  2    11 Math    <tibble [460 × 25]> m1    <S3: lm> 0.1886003
##  3    11 Rdg     <tibble [453 × 25]> m1    <S3: lm> 0.02066316
##  4    11 Science <tibble [438 × 25]> m1    <S3: lm> 0.1259080
##  5    11 Wri     <tibble [453 × 25]> m1    <S3: lm> 0.04516650
##  6     3 ELA     <tibble [540 × 25]> m1    <S3: lm> 0.04943275
##  7     3 Math    <tibble [536 × 25]> m1    <S3: lm> 0.08017486
##  8     3 Rdg     <tibble [540 × 25]> m1    <S3: lm> 0.05987155
##  9     3 Wri     <tibble [540 × 25]> m1    <S3: lm> 0.05034445
## 10     4 ELA     <tibble [585 × 25]> m1    <S3: lm> 0.1103205
## # … with 83 more rows
```

Plot

```
mods_grade %>%
    gather(model, output, m1:m3) %>%
    mutate(r2 = map_dbl(output, ~summary(.x)$r.squared)) %>%
ggplot(aes(model, r2)) +
    geom_col(aes(fill = model)) +
    facet_grid(grade ~ content) +
    guides(fill = "none") +
    scale_fill_brewer(palette = "Set2")
```


ELA
Math
Rdg
Science
Wri
r2
0.4
0.3
0.2
0.1
0.0
3
4
5
6
7
8
11
m1
m2
m3
model

etc.

Questions?

In-class Midterm