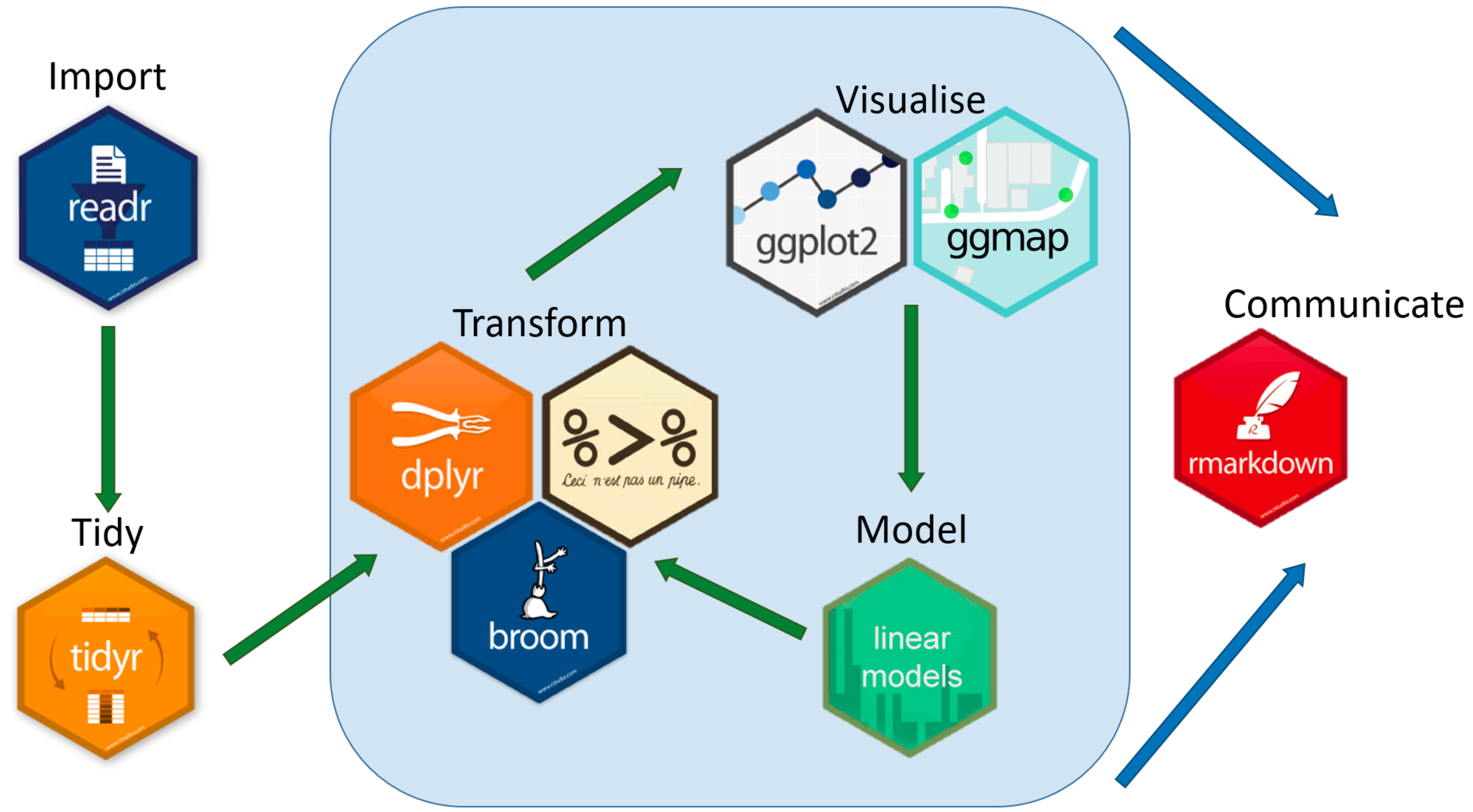
The widyr package

Pairwise correlations, clustering, and dimensionality reduction in the tidyverse

David Robinson, 2020-08-15

The tidyverse makes many data explorations fluid



Example: the gapminder dataset of country statistics

```r
library(gapminder)
gapminder
```

```
# A tibble: 1,704 x 6
   country     continent  year lifeExp      pop gdpPercap
   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
 1 Afghanistan Asia       1952    28.8  8425333      779.
 2 Afghanistan Asia       1957    30.3  9240934      821.
 3 Afghanistan Asia       1962    32.0 10267083      853.
 4 Afghanistan Asia       1967    34.0 11537966      836.
 5 Afghanistan Asia       1972    36.1 13079460      740.
 6 Afghanistan Asia       1977    38.4 14880372      786.
 7 Afghanistan Asia       1982    39.9 12881816      978.
 8 Afghanistan Asia       1987    40.8 13867957      852.
 9 Afghanistan Asia       1992    41.7 16317921      649.
10 Afghanistan Asia       1997    41.8 22227415      635.
# … with 1,694 more rows
```

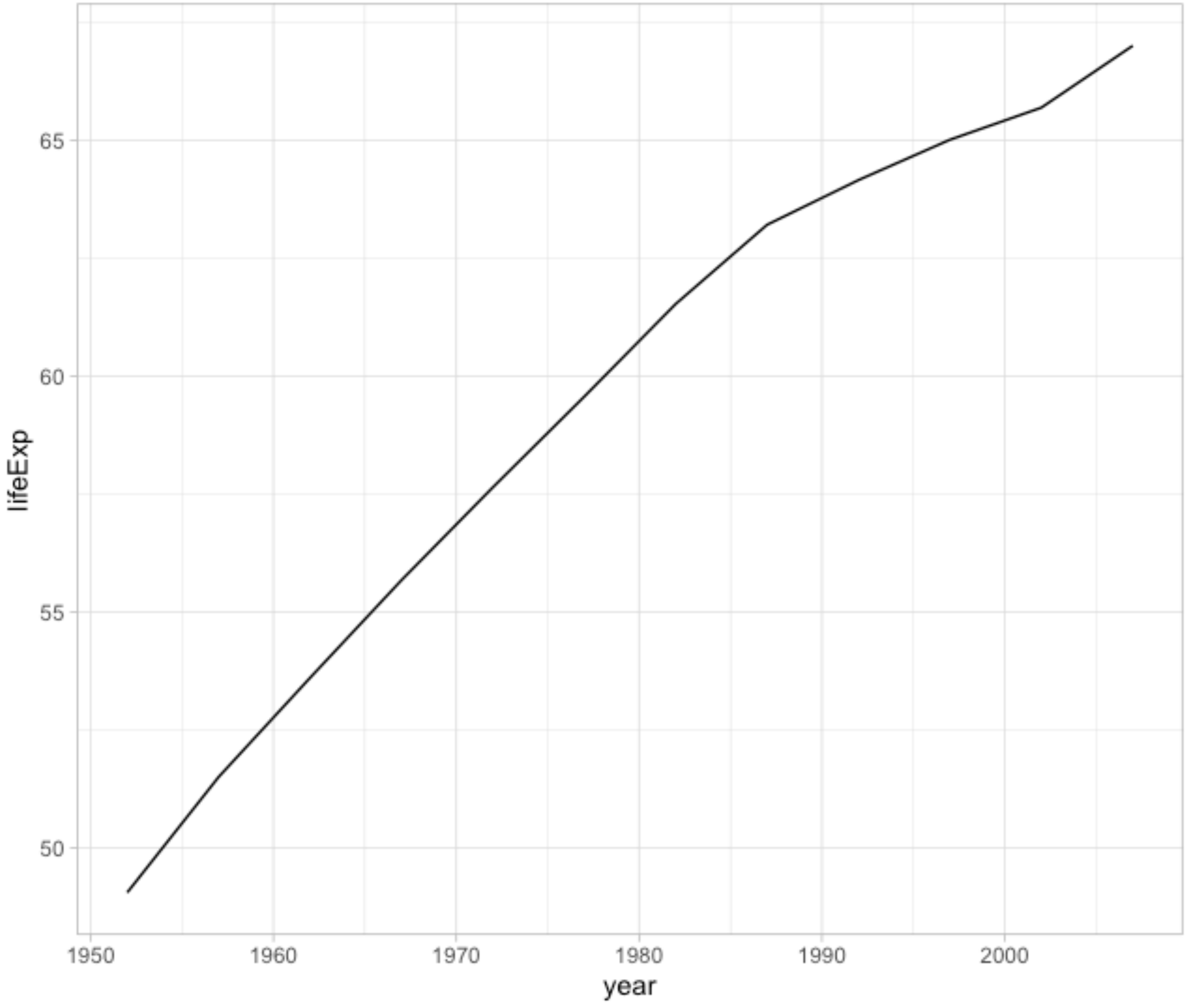
“Find the average life expectancy per year”

```
gapminder %>%
  group_by(year) %>%
  summarize(lifeExp = mean(lifeExp))
```

```
# A tibble: 12 x 2
    year lifeExp
   <int>   <dbl>
 1  1952    49.1
 2  1957    51.5
 3  1962    53.6
 4  1967    55.7
 5  1972    57.6
 6  1977    59.6
 7  1982    61.5
 8  1987    63.2
 9  1992    64.2
10  1997    65.0
11  2002    65.7
12  2007    67.0
```

“Plot the average life expectancy per year”

```
gapminder %>%
  group_by(year) %>%
  summarize(lifeExp = mean(lifeExp)) %>%
  ggplot(aes(year, lifeExp)) +
  geom_line()
```



“Find the slope of increasing life expectancy by country”

```
gapminder %>%
  group_by(country) %>%
  summarize(model = list(lm(lifeExp ~ year))) %>%
  mutate(tidied = map(model, tidy)) %>%
  unnest(tidied) %>%
  filter(term == "year")
```

```
# A tibble: 142 x 7
   country     model  term  estimate std.error statistic  p.value
   <fct>       <list> <chr>    <dbl>     <dbl>     <dbl>    <dbl>
 1 Afghanistan <lm>   year     0.275   0.0205      13.5  9.84e- 8
 2 Albania     <lm>   year     0.335   0.0332      10.1  1.46e- 6
 3 Algeria     <lm>   year     0.569   0.0221      25.7  1.81e-10
 4 Angola      <lm>   year     0.209   0.0235       8.90 4.59e- 6
 5 Argentina   <lm>   year     0.232   0.00489     47.4  4.22e-13
 6 Australia   <lm>   year     0.228   0.0104      21.9  8.67e-10
 7 Austria     <lm>   year     0.242   0.00681     35.5  7.44e-12
 8 Bahrain     <lm>   year     0.468   0.0274      17.0  1.02e- 8
 9 Bangladesh  <lm>   year     0.498   0.0163      30.5  3.37e-11
10 Belgium     <lm>   year     0.209   0.00490     42.7  1.20e-12
# … with 132 more rows
```

dplyr tidyr broom

“How is each country’s life expectancy correlated with each other?”

…

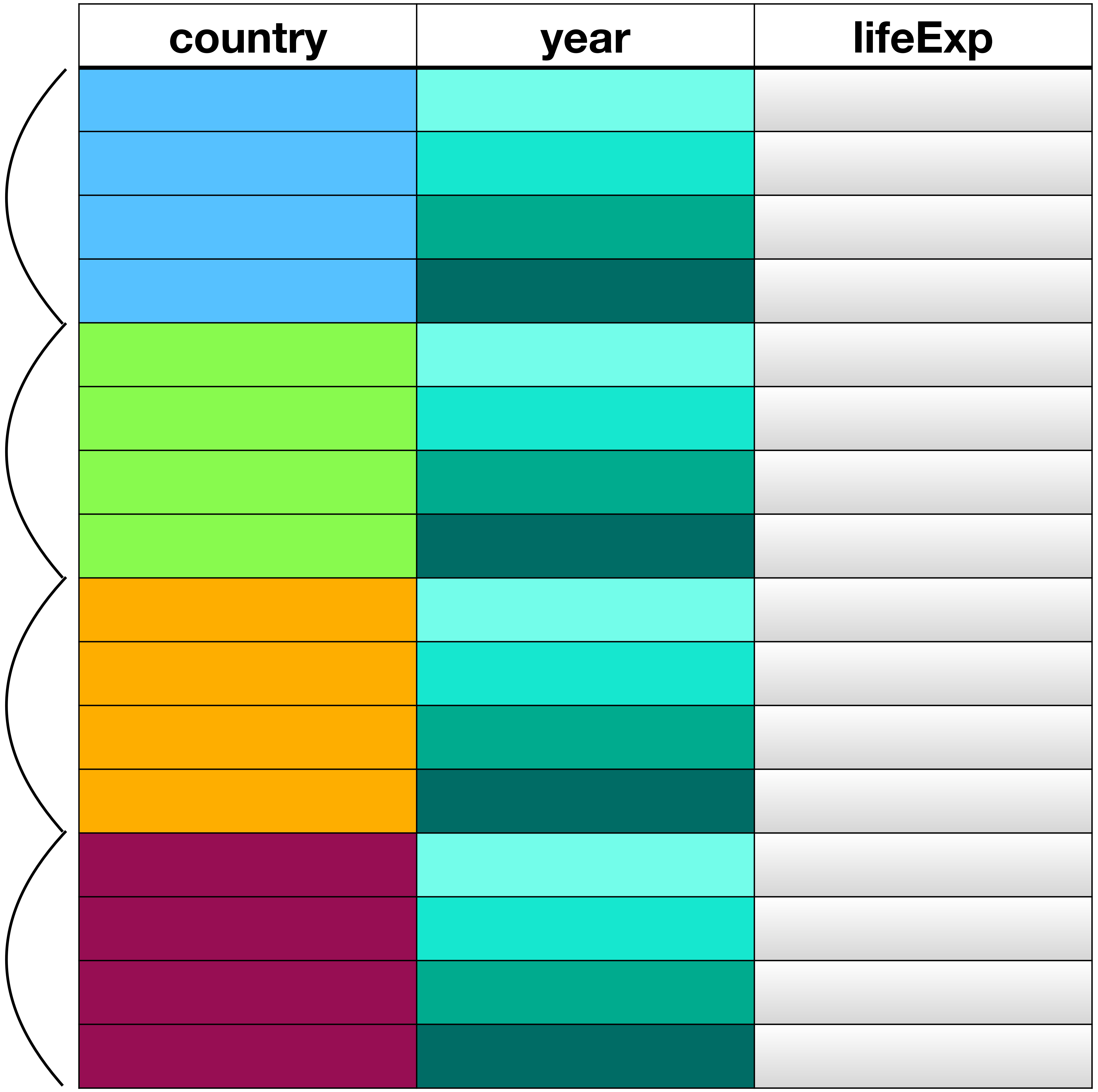
“How is each country’s life expectancy correlated with each other?”

```r
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

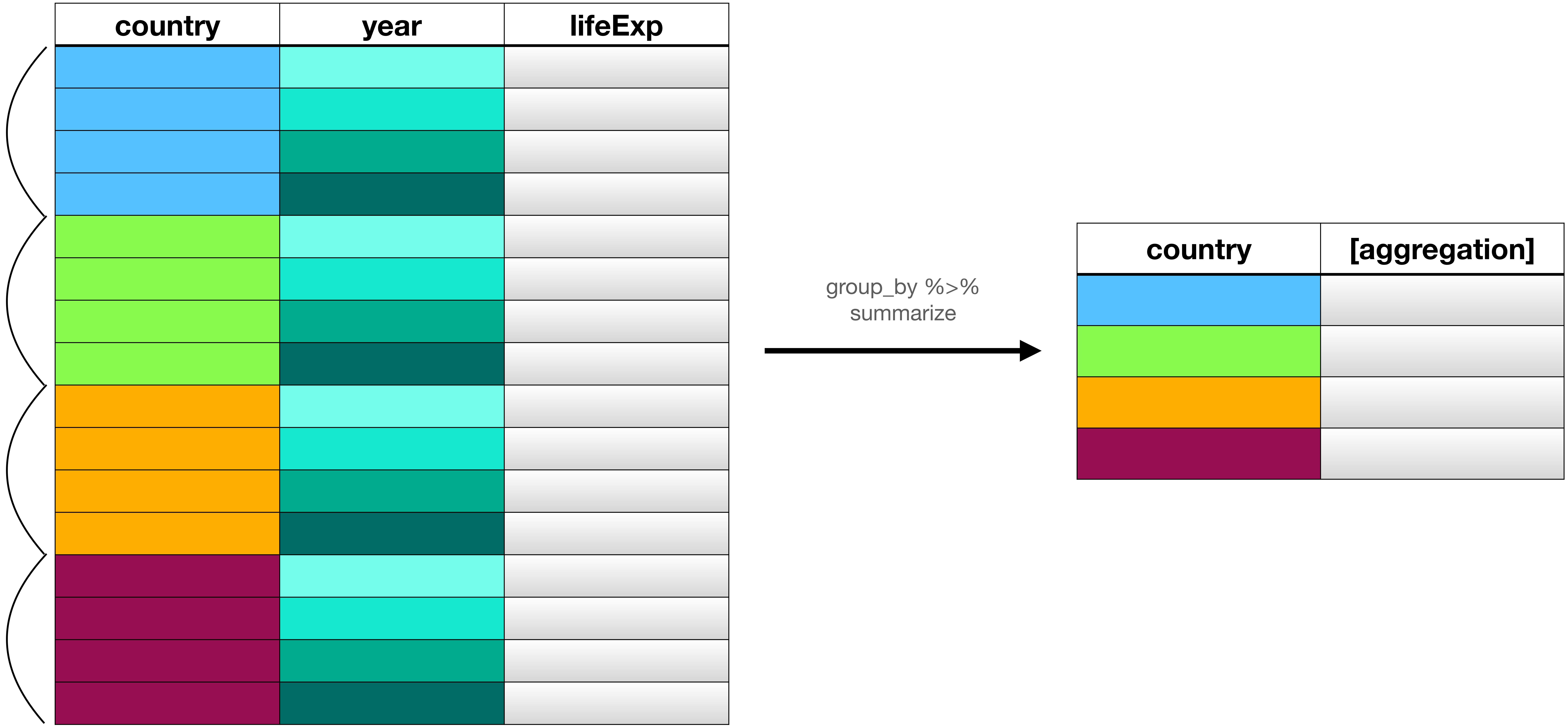
```
# A tibble: 20,022 x 3
   item1              item2              correlation
   <fct>              <fct>                    <dbl>
 1 Mauritania         Indonesia                 1.00
 2 Indonesia          Mauritania                1.00
 3 Senegal            Morocco                   1.00
 4 Morocco            Senegal                   1.00
 5 West Bank and Gaza Saudi Arabia              1.00
 6 Saudi Arabia       West Bank and Gaza        1.00
 7 France             Brazil                   0.999
 8 Brazil             France                   0.999
 9 Reunion            Bahrain                  0.999
10 Bahrain            Reunion                  0.999
# … with 20,012 more rows
```

How pairwise operations work

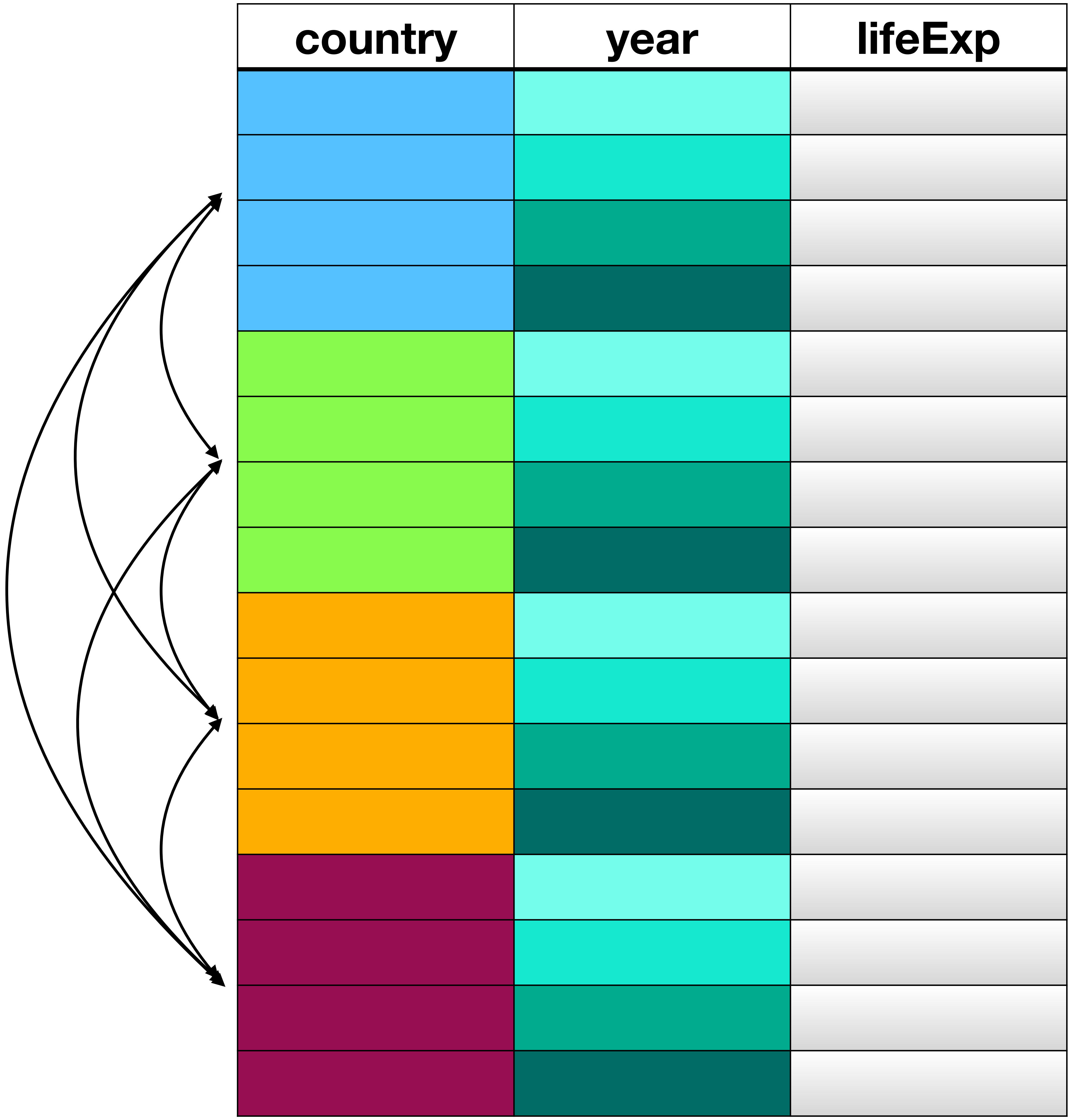
dplyr is well suited for “aggregate within groups”



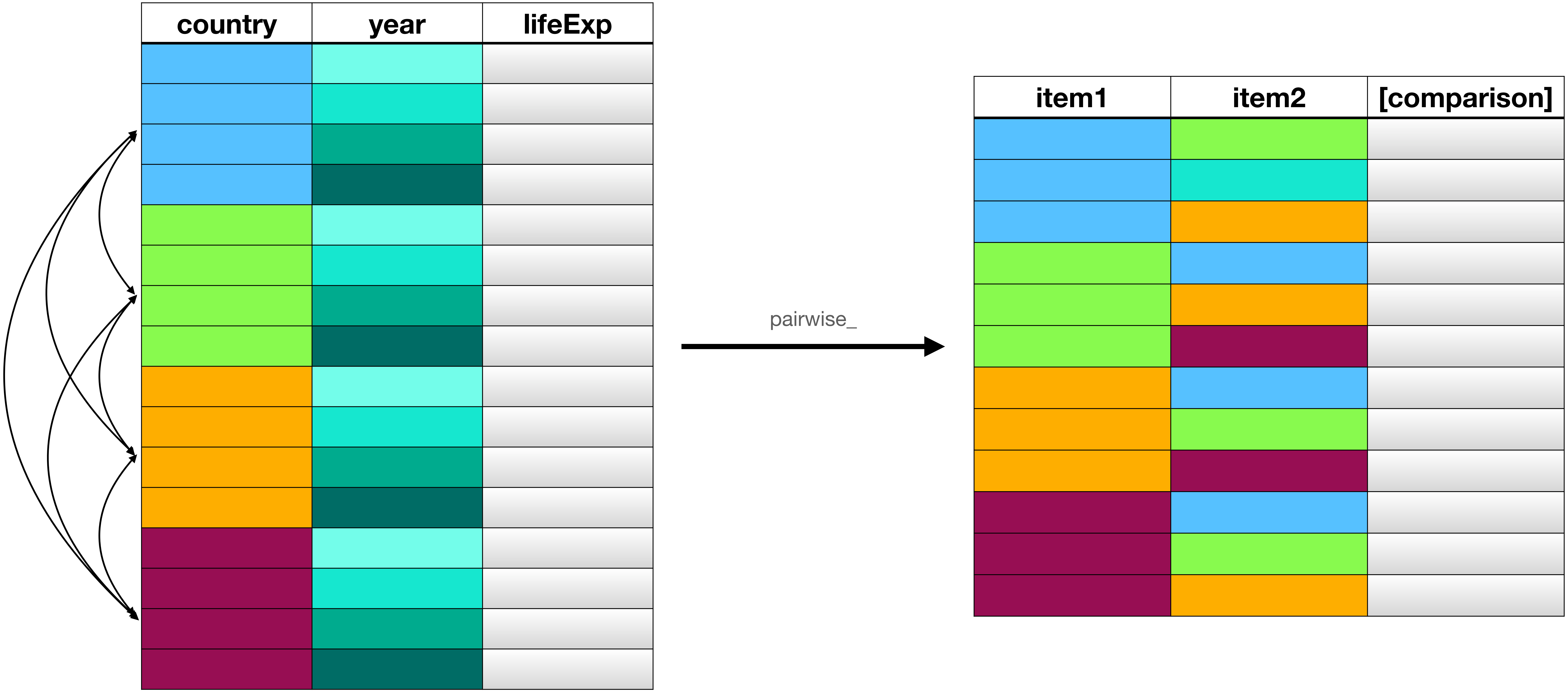
dplyr is well suited for “aggregate within groups”



pairwise_ operations compare each *pair* of items



pairwise_ operations compare each *pair* of items



Correlations in R are traditionally done on matrices

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
[1,]	39.1	18.7	181	3750
[2,]	39.5	17.4	186	3800
[3,]	40.3	18.0	195	3250
[4,]	36.7	19.3	193	3450
[5,]	39.3	20.6	190	3650
[6,]	38.9	17.8	181	3625
[7,]	39.2	19.6	195	4675
[8,]	41.1	17.6	182	3200
[9,]	38.6	21.2	191	3800
[10,]	34.6	21.1	198	4400
[11,]	36.6	17.8	185	3700
[12,]	38.7	19.0	195	3450
[13,]	42.5	20.7	197	4500
[14,]	34.4	18.4	184	3325
[15,]	46.0	21.5	194	4200
[16,]	37.8	18.3	174	3400
[17,]	37.7	18.7	180	3600
[18,]	35.9	19.2	189	3800
[19,]	38.2	18.1	185	3950
[20,]	38.8	17.2	180	3800

```
cor(penguin_matrix)
```

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
bill_length_mm	1.0000000	-0.2286256	0.6530956	0.5894511
bill_depth_mm	-0.2286256	1.0000000	-0.5777917	-0.4720157
flipper_length_mm	0.6530956	-0.5777917	1.0000000	0.8729789
body_mass_g	0.5894511	-0.4720157	0.8729789	1.0000000

Me working with any data format that's not a tidy table

The widen-operate-retidy pattern

```
gapminder %>%
  select(country, year, lifeExp) %>%
  pivot_wider(names_from = country, values_from = lifeExp) %>%
  select(-year) %>%
  cor(use = "pairwise.complete.obs") %>%
  as_tibble(rownames = "item1") %>%
  pivot_longer(cols = -item1, names_to = "item2")
```

```
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

The widen-operate-retidy pattern

```
gapminder %>%
  select(country, year, lifeExp) %>%
  pivot_wider(names_from = country, values_from = lifeExp) %>%
  select(-year) %>%
  cor(use = "pairwise.complete.obs") %>%
  as_tibble(rownames = "item1") %>%
  pivot_longer(cols = -item1, names_to = "item2")
```

Widen

```
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

The widen-operate-retidy pattern

```
gapminder %>%
  select(country, year, lifeExp) %>%
  pivot_wider(names_from = country, values_from = lifeExp) %>%
  select(-year) %>%
  cor(use = "pairwise.complete.obs") %>%
  as_tibble(rownames = "item1") %>%
  pivot_longer(cols = -item1, names_to = "item2")
```

Operate

```
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

The widen-operate-retidy pattern

```
gapminder %>%
  select(country, year, lifeExp) %>%
  pivot_wider(names_from = country, values_from = lifeExp) %>%
  select(-year) %>%
  cor(use = "pairwise.complete.obs") %>%
  as_tibble(rownames = "item1") %>%
  pivot_longer(cols = -item1, names_to = "item2")
```

Re-tidy

```
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

The widen-operate-retidy pattern

```r
gapminder %>%
  select(country, year, lifeExp) %>%
  pivot_wider(names_from = country, values_from = lifeExp) %>%
  select(-year) %>%
  cor(use = "pairwise.complete.obs") %>%
  as_tibble(rownames = "item1") %>%
  pivot_longer(cols = -item1, names_to = "item2")
```

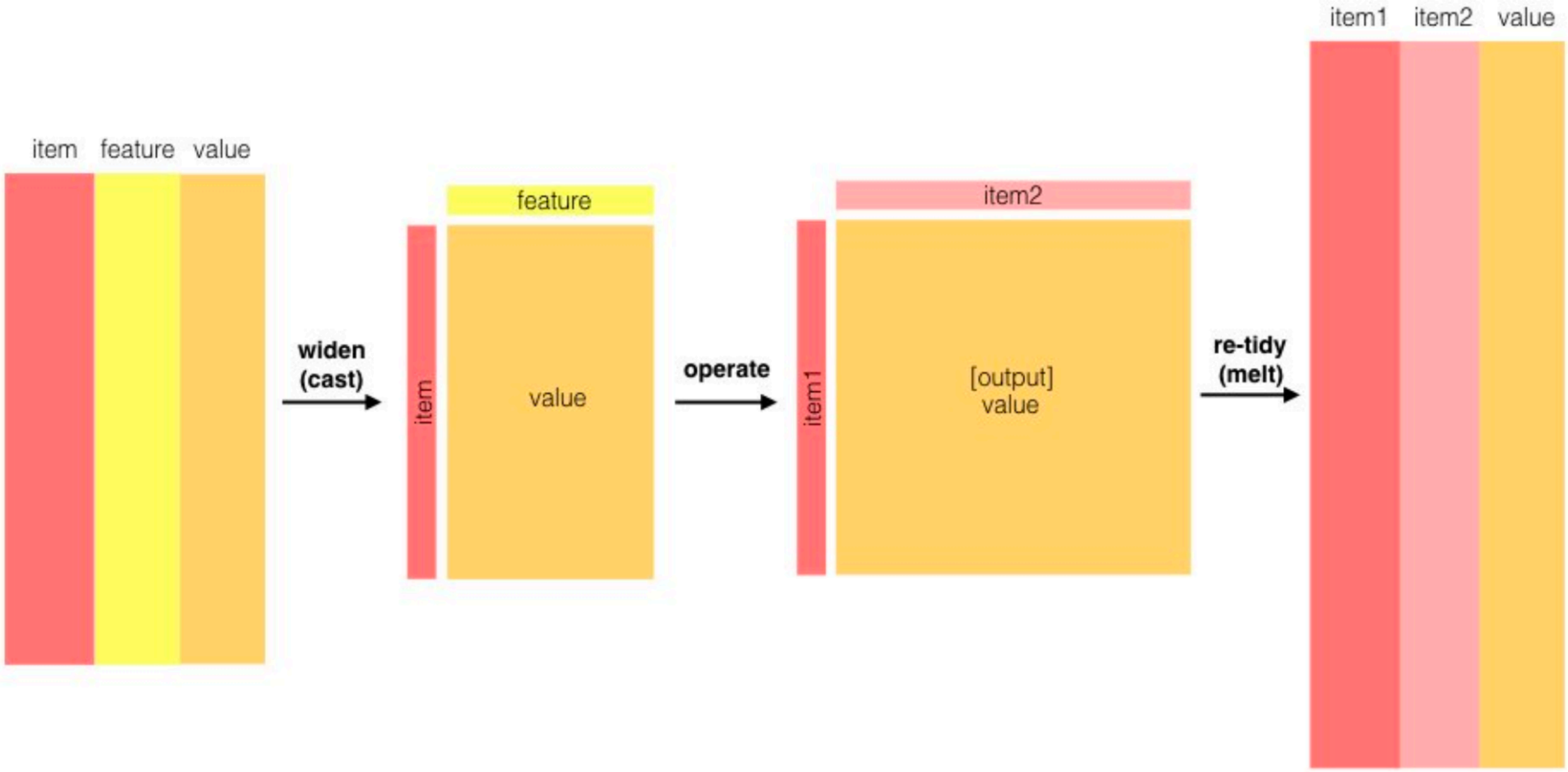
Widen

Operate

Re-tidy

```r
library(widyr)

gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```

The widen-operate-retidy pattern

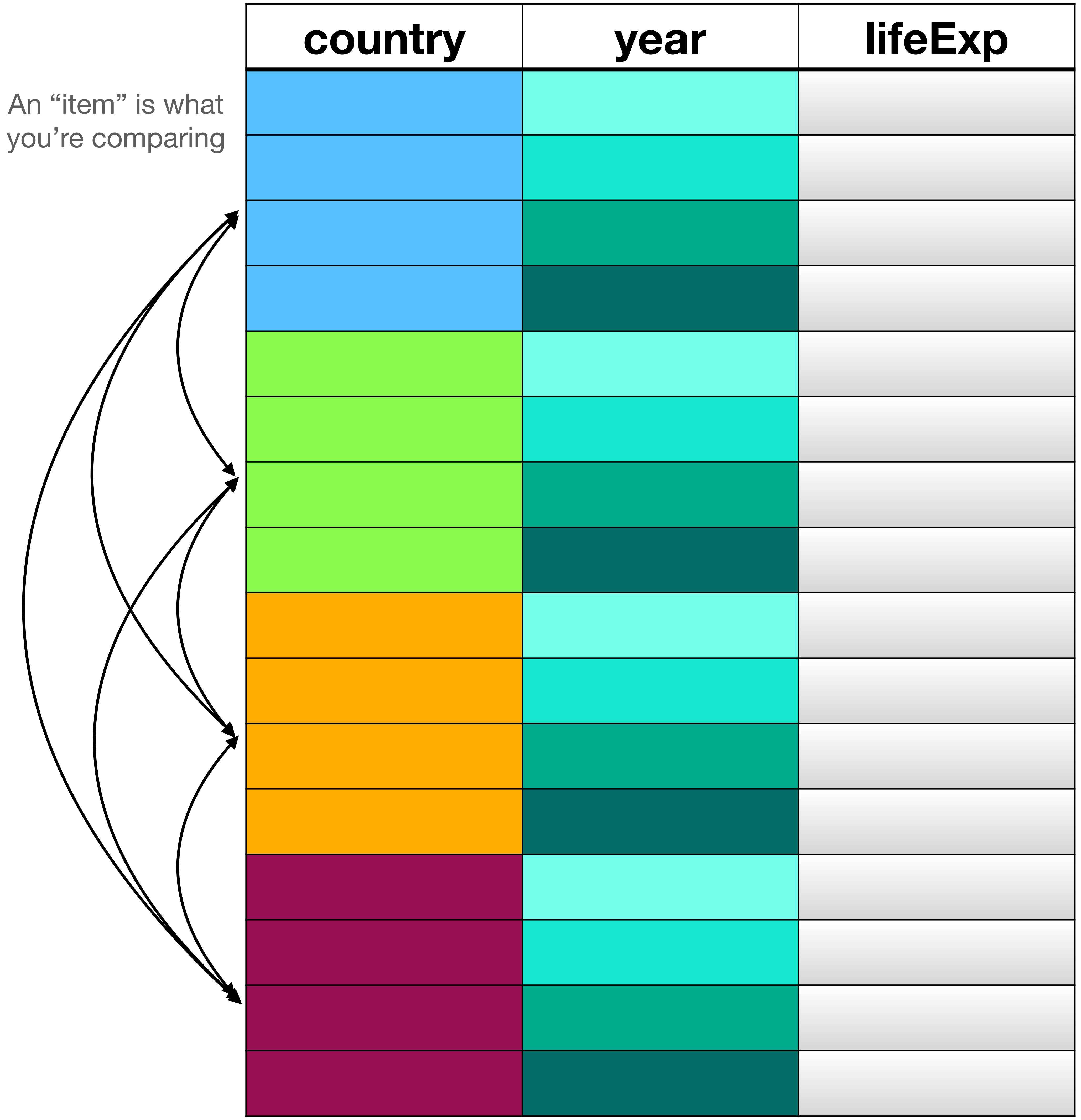


Input is tidy

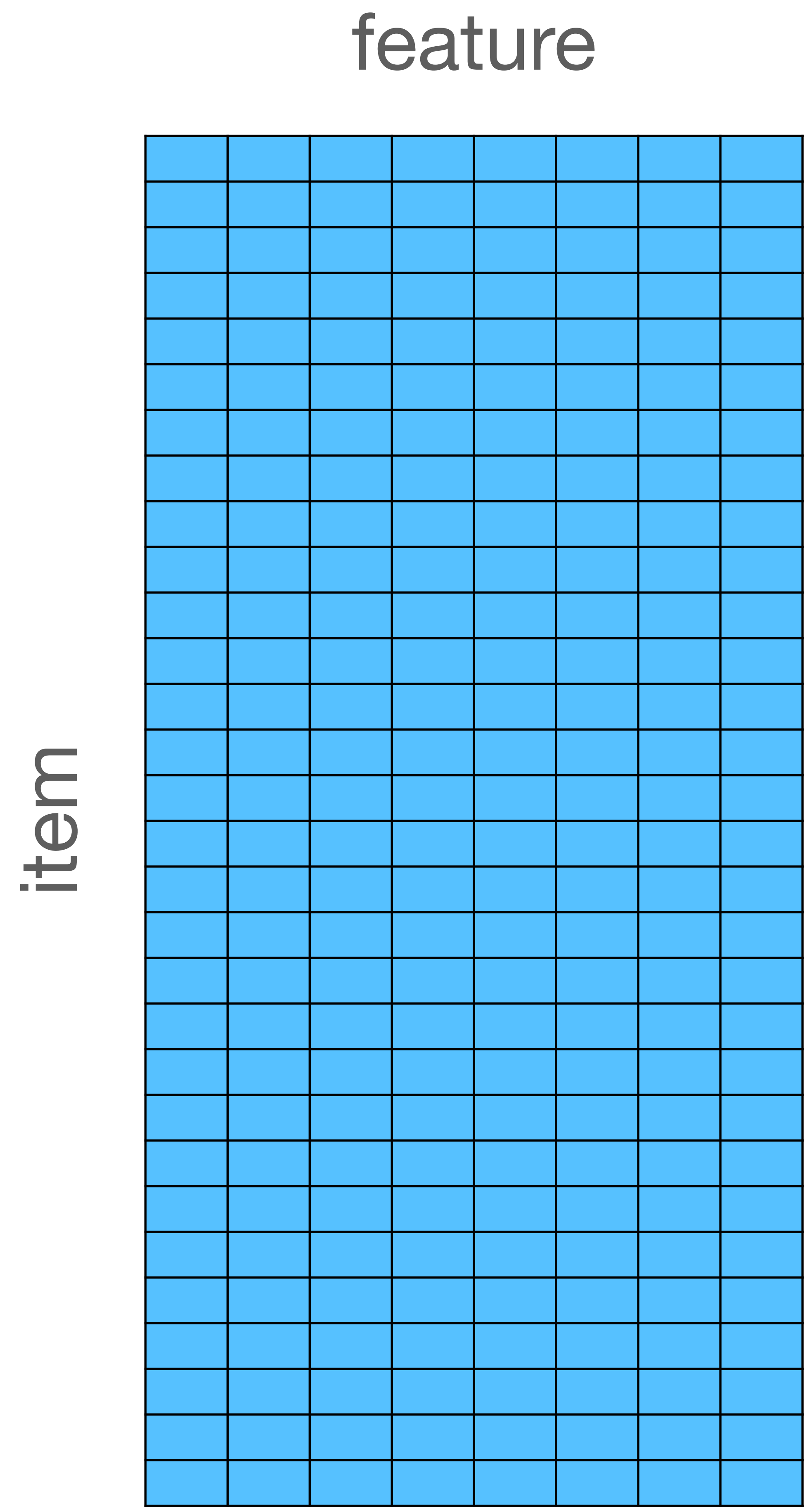
Middle is "wide" matrices

Output is tidy

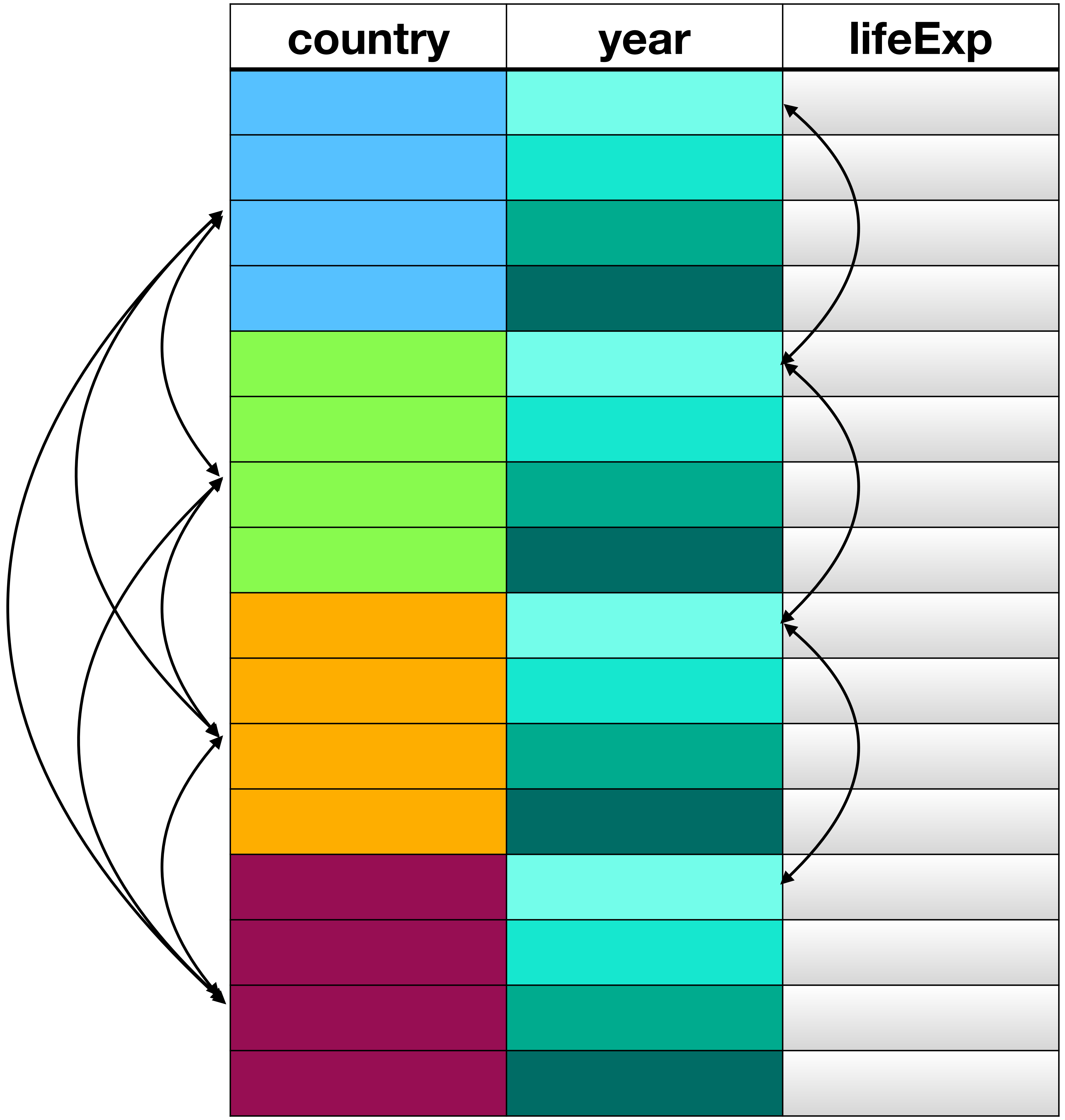
pairwise_ operations compares pairs of *items*



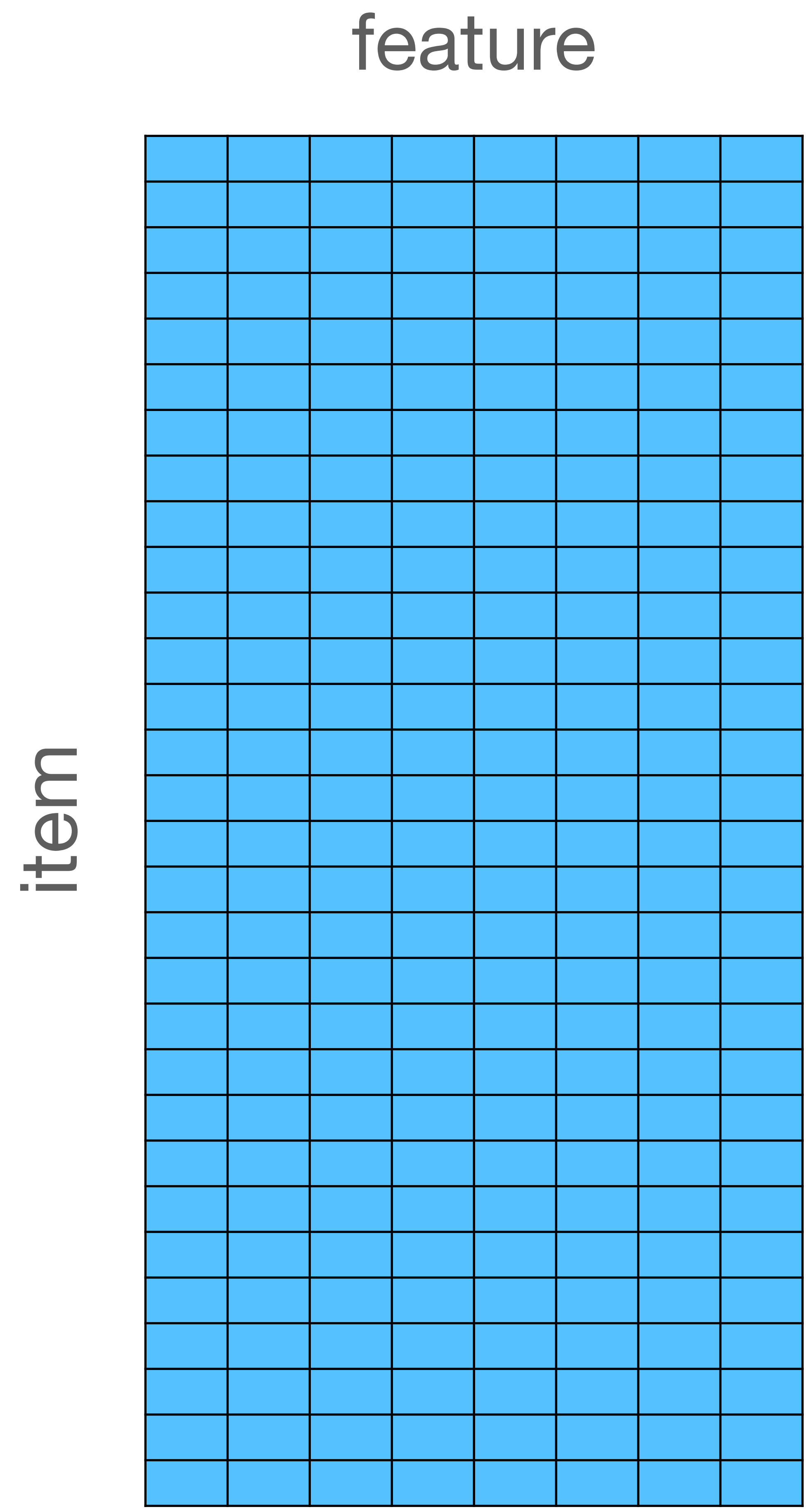

```r
gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```



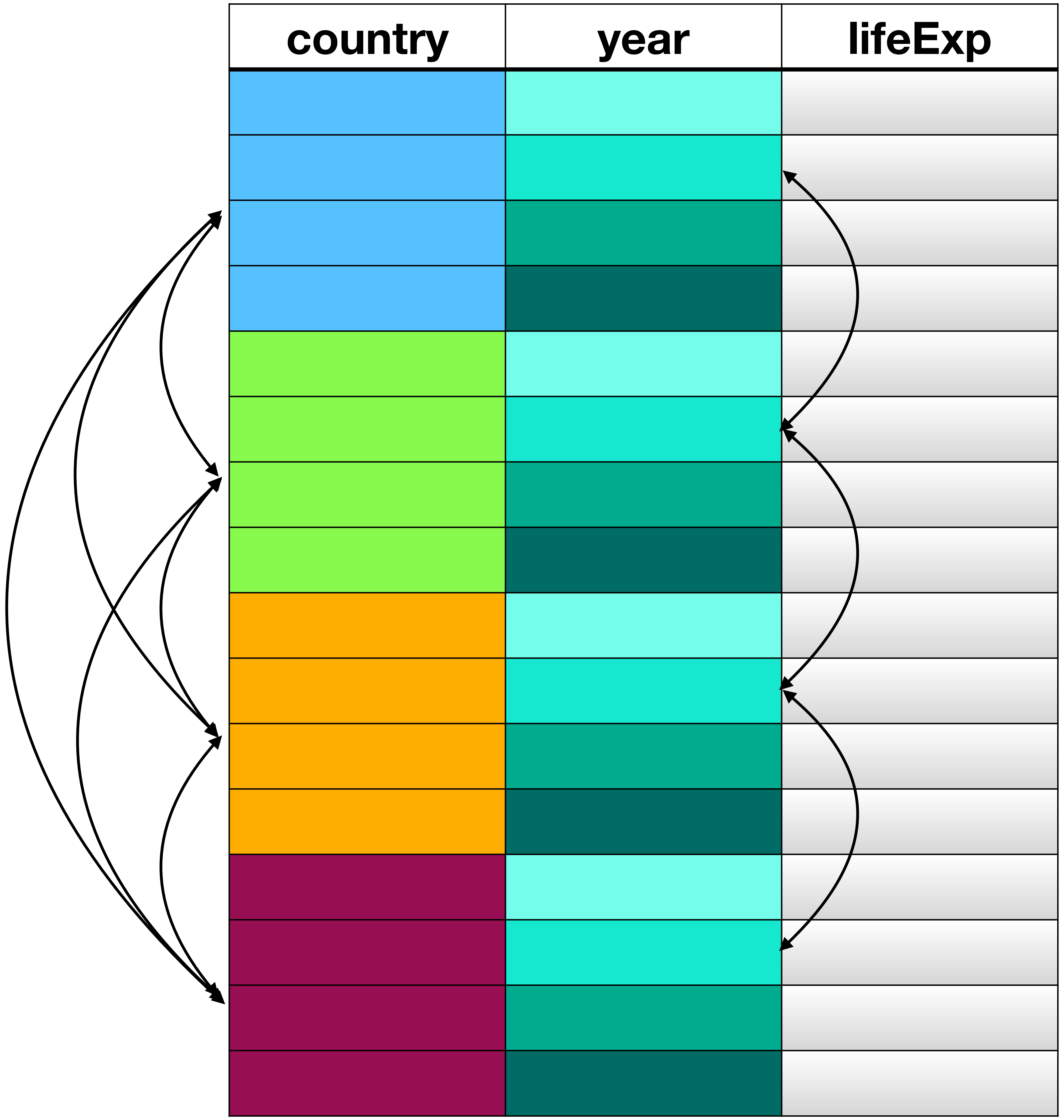
A *feature* is the second dimension, that links observations together



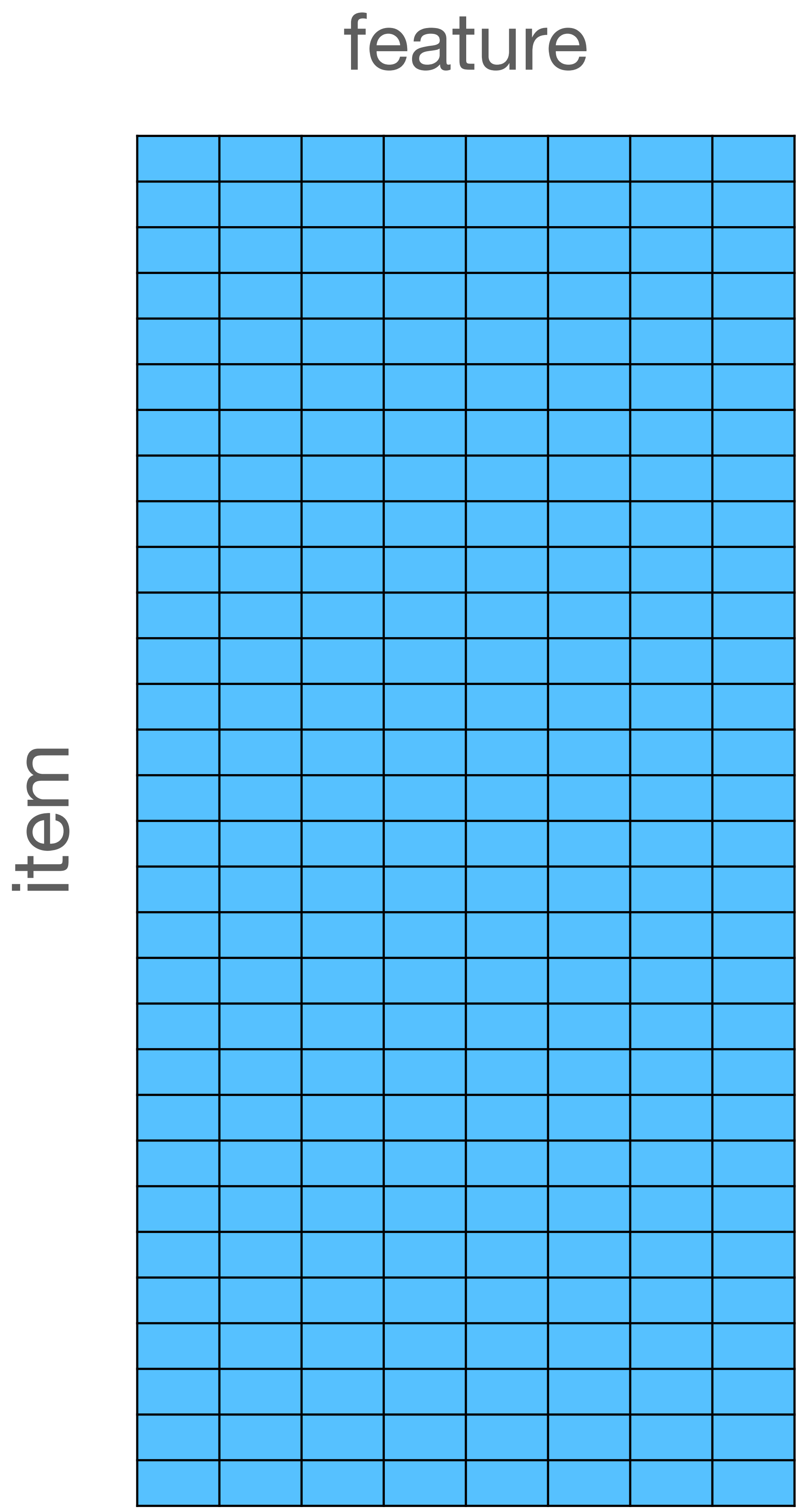

```
gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```



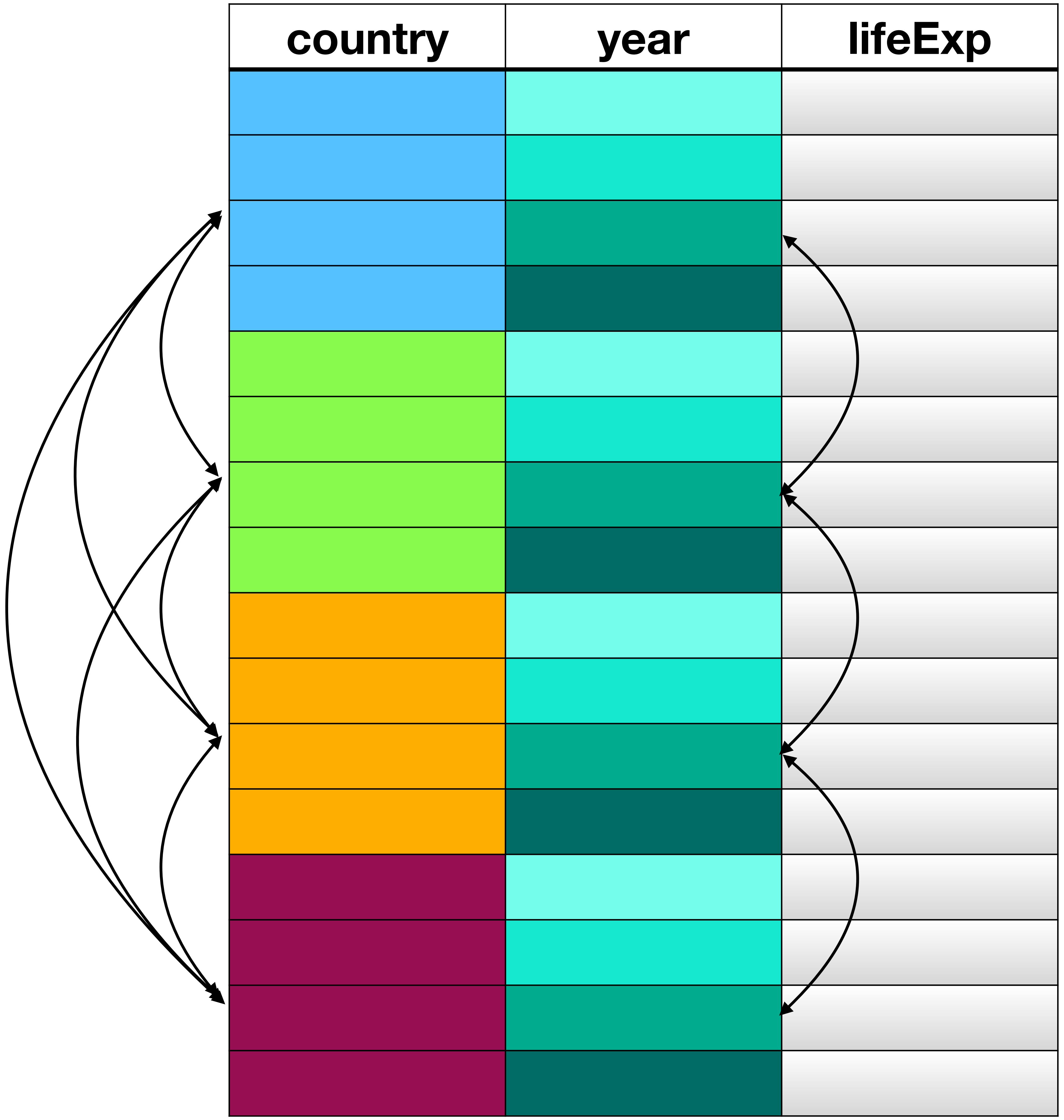
A *feature* is the second dimension, that links items together



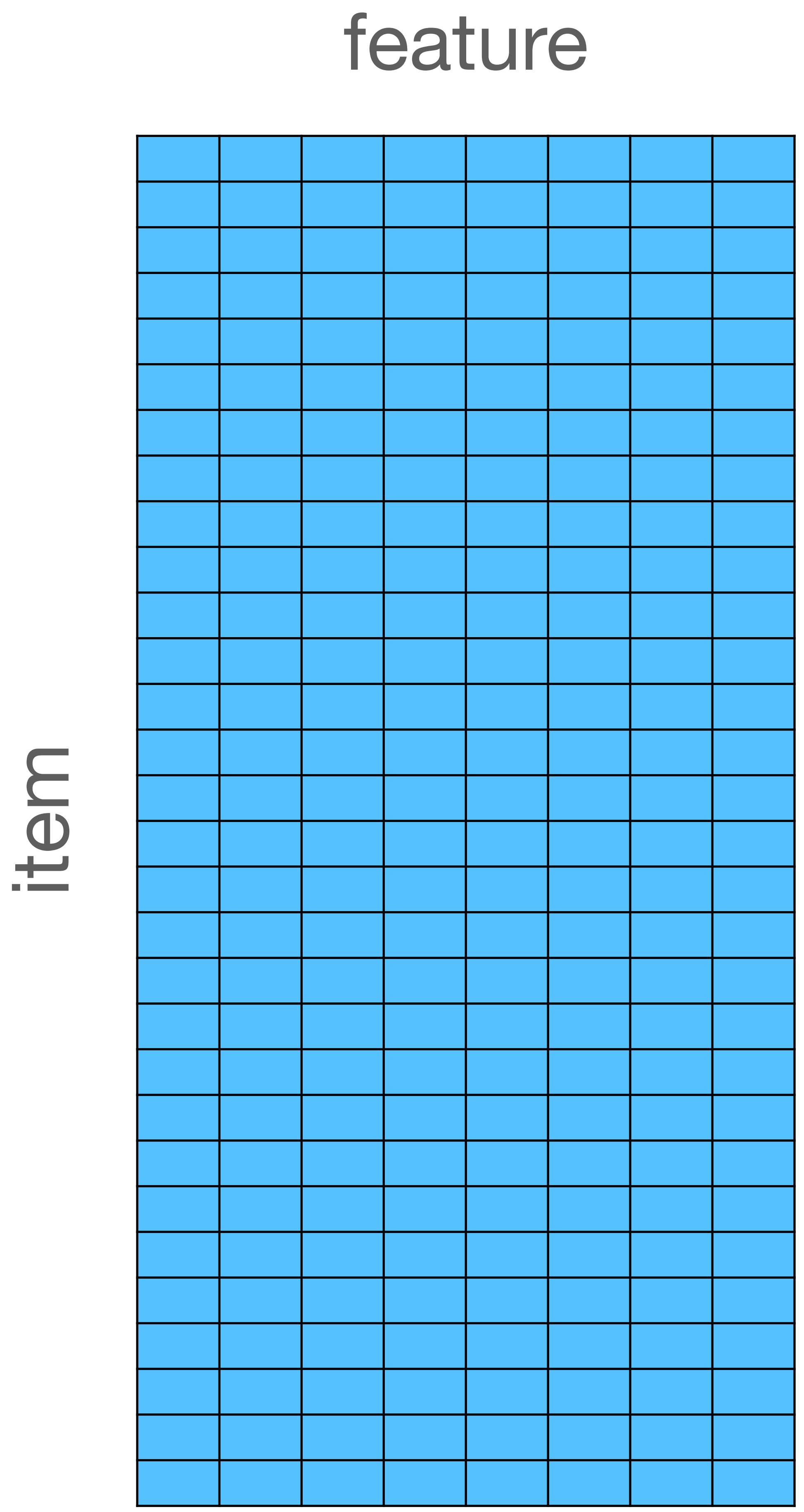

```
gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```



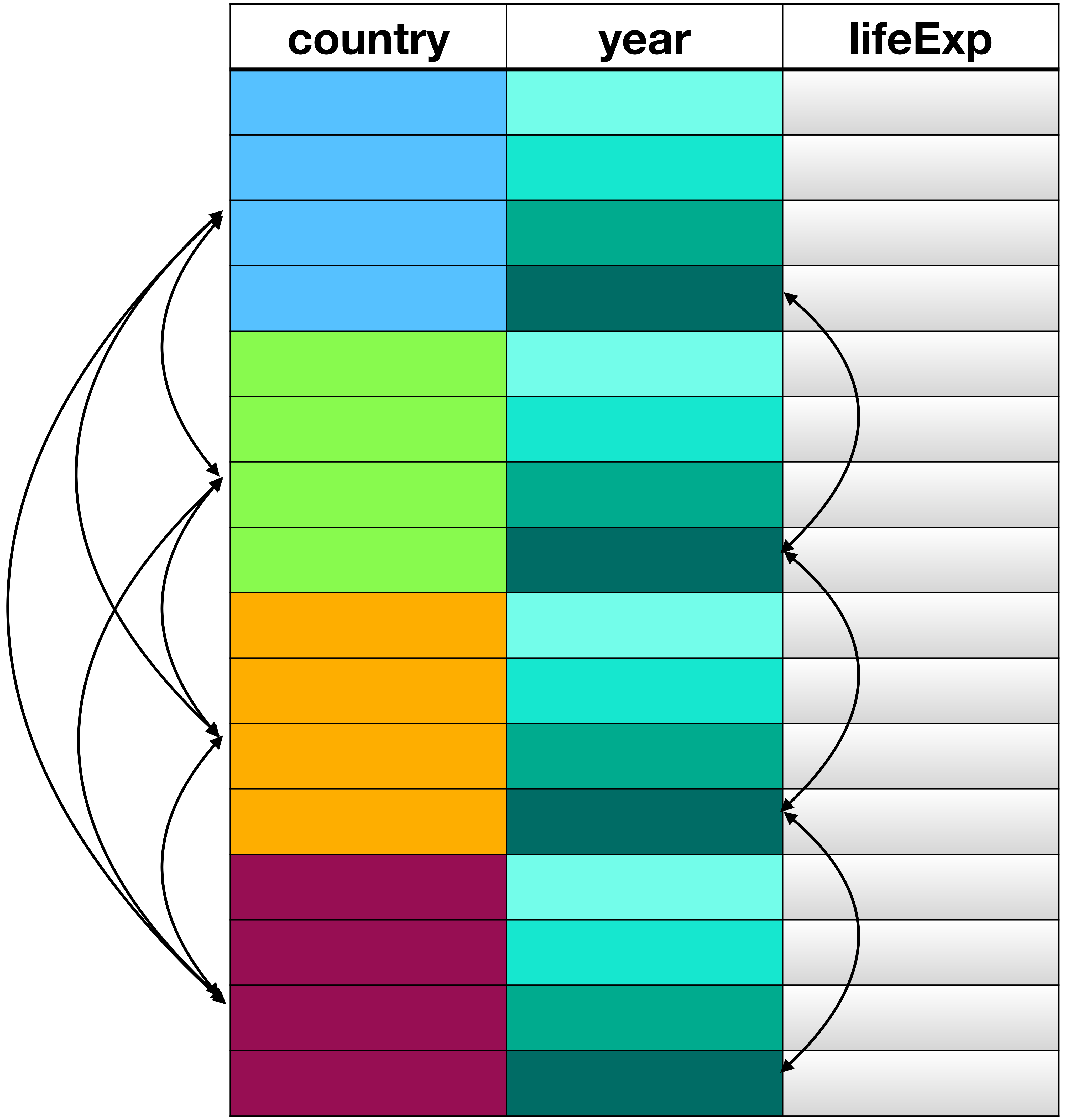
A *feature* is the second dimension, that links items together



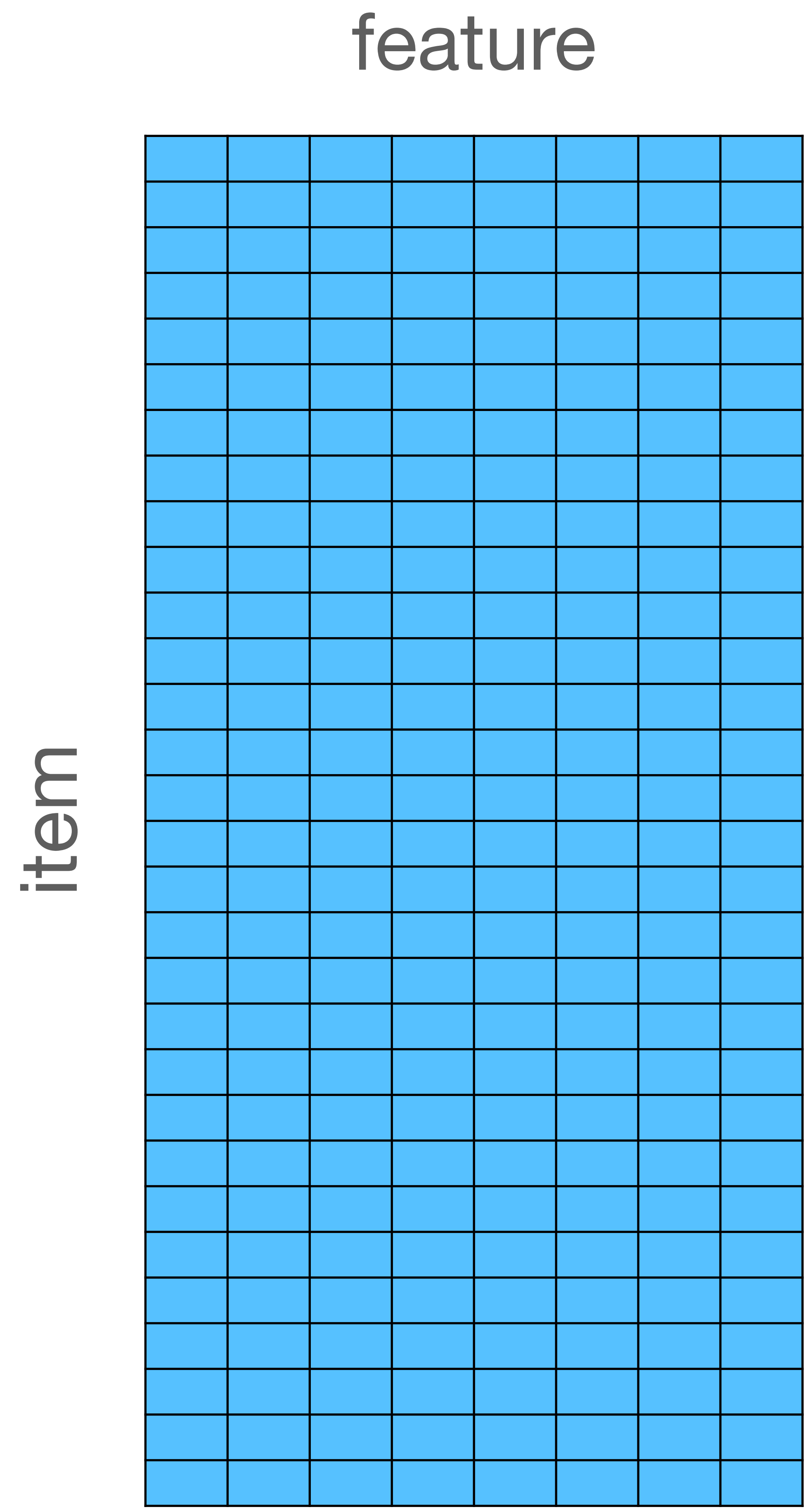

```
gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```



A *feature* is the second dimension, that links items together




```r
gapminder %>%
  pairwise_cor(country, year, lifeExp, sort = TRUE)
```



Pairwise example:
United Nations voting

United Nations voting data

```r
library(unvotes)
```

```
# A tibble: 733,404 x 4
    rcid country                  country_code  vote
   <int> <chr>                    <chr>        <dbl>
 1     3 United States of America US               1
 2     3 Canada                   CA              -1
 3     3 Cuba                     CU               1
 4     3 Haiti                    HT               1
 5     3 Dominican Republic       DO               1
 6     3 Mexico                   MX               1
 7     3 Guatemala                GT               1
 8     3 Honduras                 HN               1
 9     3 El Salvador              SV               1
10     3 Nicaragua                NI               1
# … with 733,394 more rows
```

United Nations voting data

```r
library(unvotes)
```

```
# A tibble: 733,404 x 4
    rcid country                  country_code  vote
   <int> <chr>                    <chr>        <dbl>
 1     3 United States of America US               1
 2     3 Canada                   CA              -1
 3     3 Cuba                     CU               1
 4     3 Haiti                    HT               1
 5     3 Dominican Republic       DO               1
 6     3 Mexico                   MX               1
 7     3 Guatemala                GT               1
 8     3 Honduras                 HN               1
 9     3 El Salvador              SV               1
10     3 Nicaragua                NI               1
# … with 733,394 more rows
```

1: Yes
0: Abstain
-1: No

United Nations voting data

```r
library(unvotes)
```

```
# A tibble: 733,404 x 4
    rcid country                  country_code  vote
   <int> <chr>                    <chr>        <dbl>
 1     3 United States of America US               1
 2     3 Canada                   CA              -1
 3     3 Cuba                     CU               1
 4     3 Haiti                    HT               1
 5     3 Dominican Republic       DO               1
 6     3 Mexico                   MX               1
 7     3 Guatemala                GT               1
 8     3 Honduras                 HN               1
 9     3 El Salvador              SV               1
10     3 Nicaragua                NI               1
# … with 733,394 more rows
```

Roll call ID (rcid) is our "feature":
How we know which pairs of votes to compare

What countries agree/disagree with each other?


```
# A tibble: 733,404 x 4
    rcid country                  country_code  vote
   <int> <chr>                    <chr>        <dbl>
 1     3 United States of America US               1
 2     3 Canada                   CA              -1
 3     3 Cuba                     CU               1
 4     3 Haiti                    HT               1
 5     3 Dominican Republic       DO               1
 6     3 Mexico                   MX               1
 7     3 Guatemala                GT               1
 8     3 Honduras                 HN               1
 9     3 El Salvador              SV               1
10     3 Nicaragua                NI               1
# … with 733,394 more rows
```

Pairwise correlations of votes

```r
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE)
```

```
# A tibble: 38,612 x 3
   item1          item2          correlation
   <chr>          <chr>                <dbl>
 1 Slovakia       Czech Republic       0.989
 2 Czech Republic Slovakia             0.989
 3 Lithuania      Estonia              0.971
 4 Estonia        Lithuania            0.971
 5 Lithuania      Latvia               0.970
 6 Latvia         Lithuania            0.970
 7 Germany        Liechtenstein        0.968
 8 Liechtenstein  Germany              0.968
 9 Slovakia       Slovenia             0.966
10 Slovenia       Slovakia             0.966
# … with 38,602 more rows
```

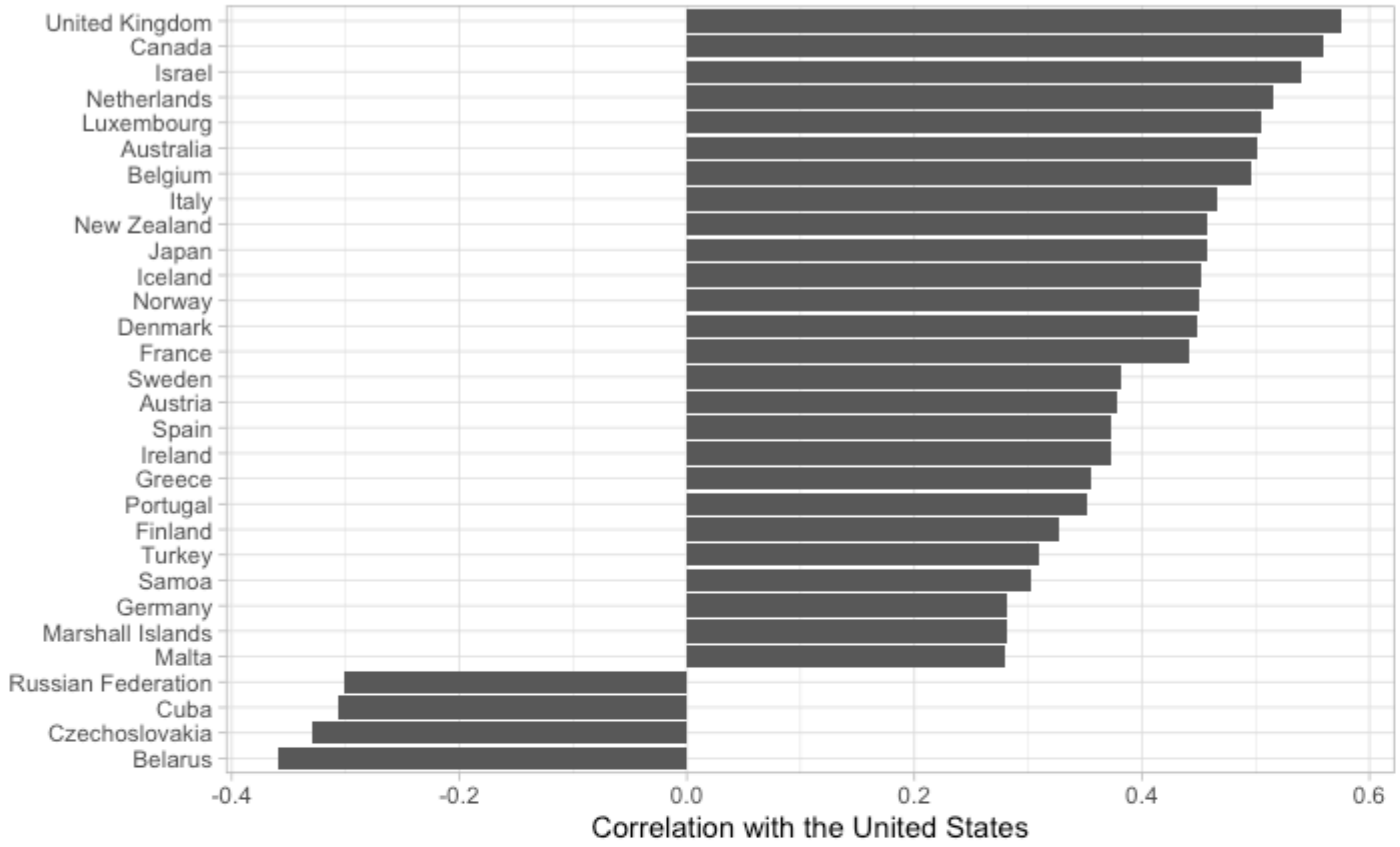
Pairwise correlations with the United States

```r
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 == "United States of America")
```

```
# A tibble: 196 x 3
   item1                    item2          correlation
   <chr>                    <chr>                <dbl>
 1 United States of America United Kingdom       0.576
 2 United States of America Canada               0.559
 3 United States of America Israel               0.540
 4 United States of America Netherlands          0.515
 5 United States of America Luxembourg           0.505
 6 United States of America Australia            0.502
 7 United States of America Belgium              0.496
 8 United States of America Italy                0.467
 9 United States of America New Zealand          0.458
10 United States of America Japan                0.458
# … with 186 more rows
```

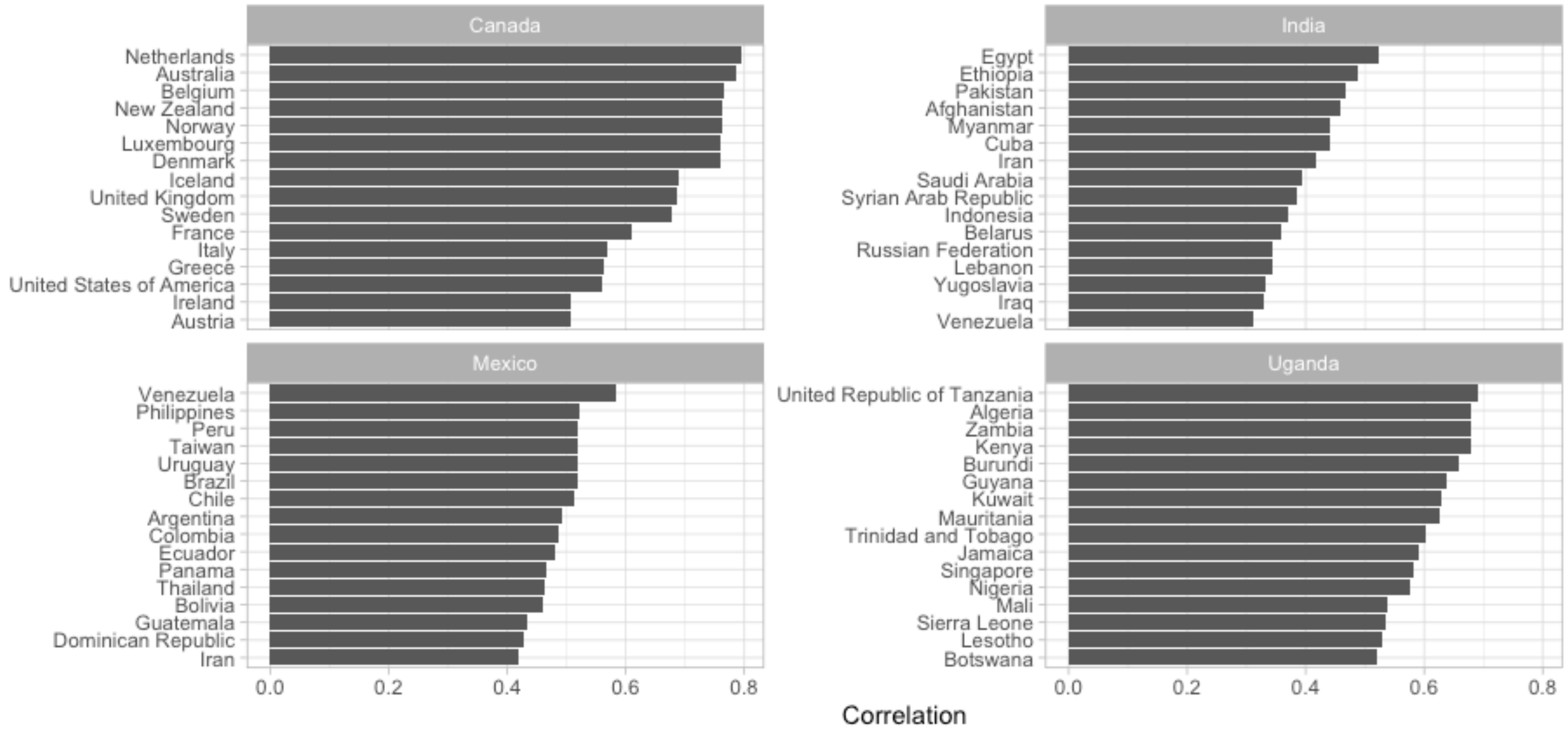
Highest/lowest correlations with the United States

```
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 == "United States of America") %>%
  top_n(30, abs(correlation)) %>%
  ggplot(aes(correlation, reorder(item2, correlation))) +
  geom_col() +
  labs(x = "Correlation with the United States", y = "")
```



Highest correlations faceted by country

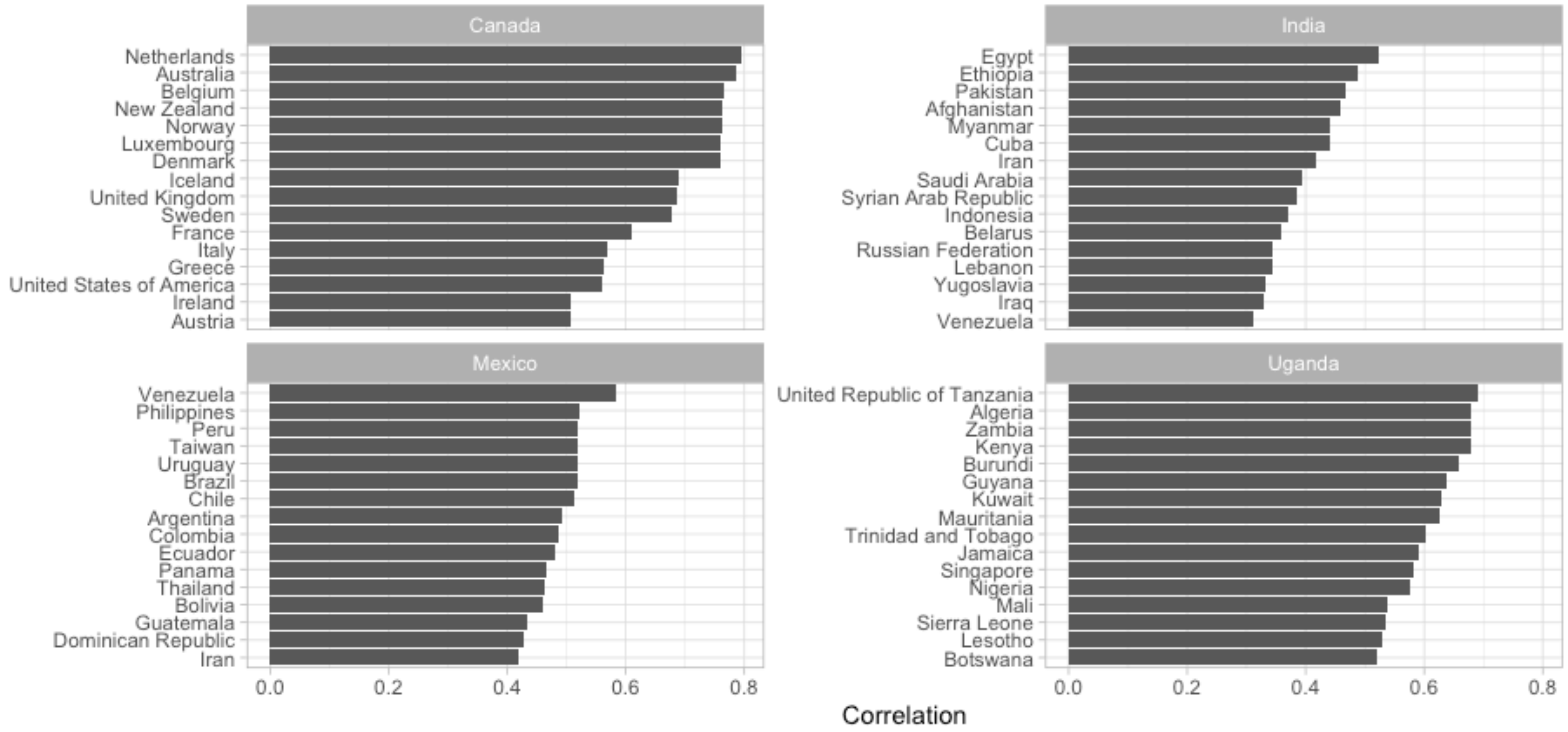
```
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 %in% c("Uganda", "India", "Canada", "Mexico")) %>%
  group_by(item1) %>%
  top_n(16, abs(correlation)) %>%
  mutate(item2 = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(correlation, item2)) +
  geom_col() +
  facet_wrap(~ item1, scales = "free_y") +
  scale_y_reordered() +
  labs(x = "Correlation", y = "")
```



Highest correlations faceted by country

widyr

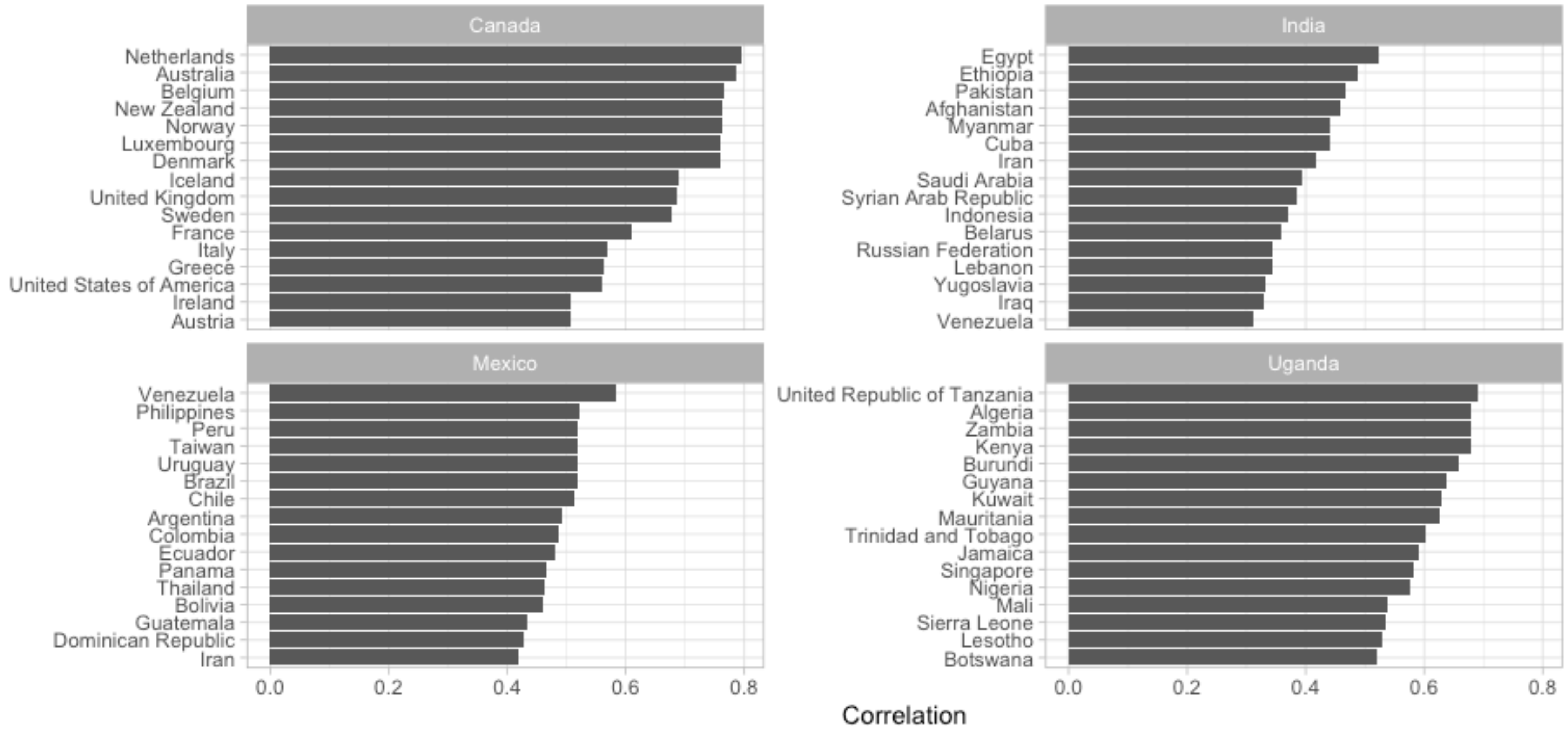
```
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 %in% c("Uganda", "India", "Canada", "Mexico")) %>%
  group_by(item1) %>%
  top_n(16, abs(correlation)) %>%
  mutate(item2 = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(correlation, item2)) +
  geom_col() +
  facet_wrap(~ item1, scales = "free_y") +
  scale_y_reordered() +
  labs(x = "Correlation", y = "")
```



Highest correlations faceted by country

dplyr

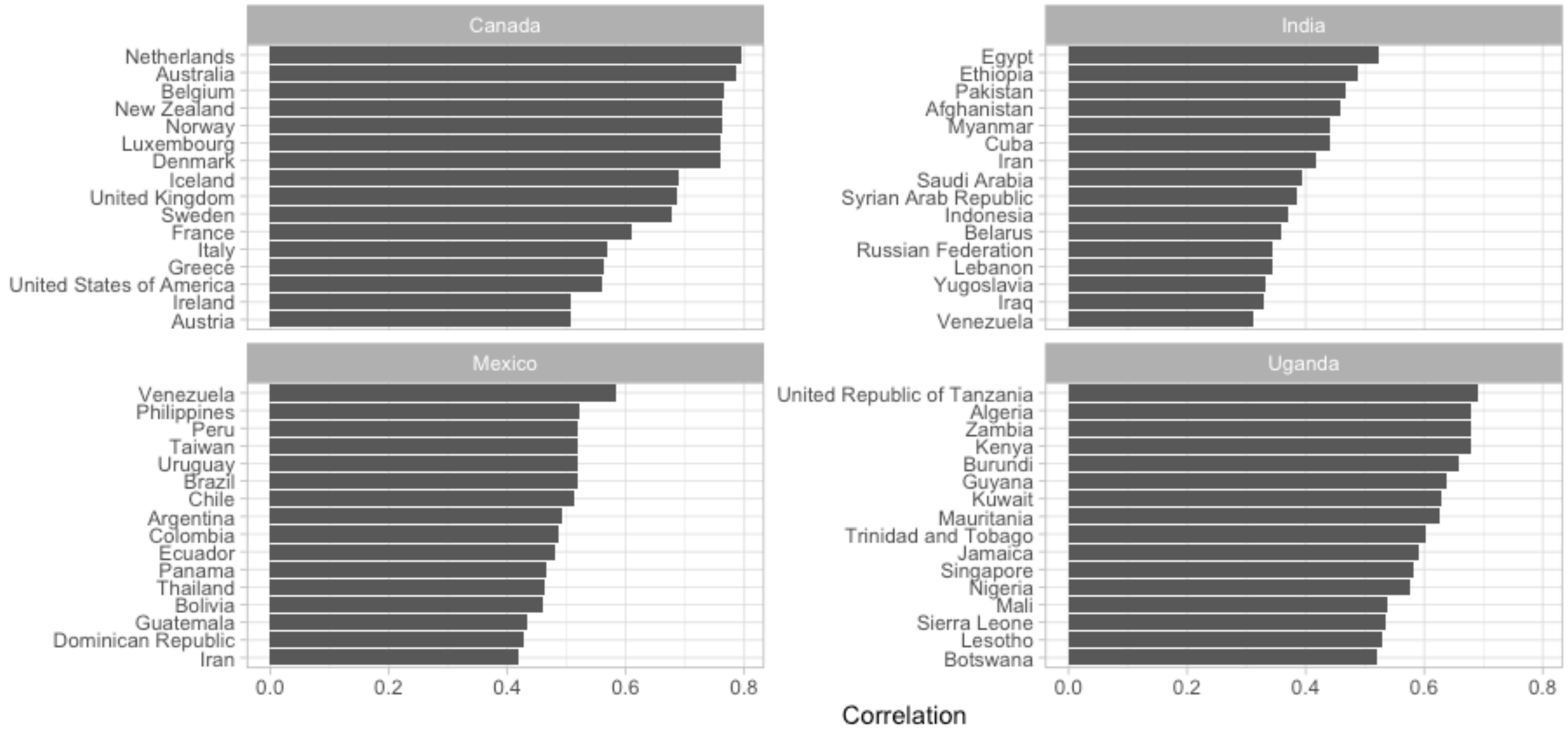
```
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 %in% c("Uganda", "India", "Canada", "Mexico")) %>%
  group_by(item1) %>%
  top_n(16, abs(correlation)) %>%
  mutate(item2 = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(correlation, item2)) +
  geom_col() +
  facet_wrap(~ item1, scales = "free_y") +
  scale_y_reordered() +
  labs(x = "Correlation", y = "")
```



Highest correlations faceted by country

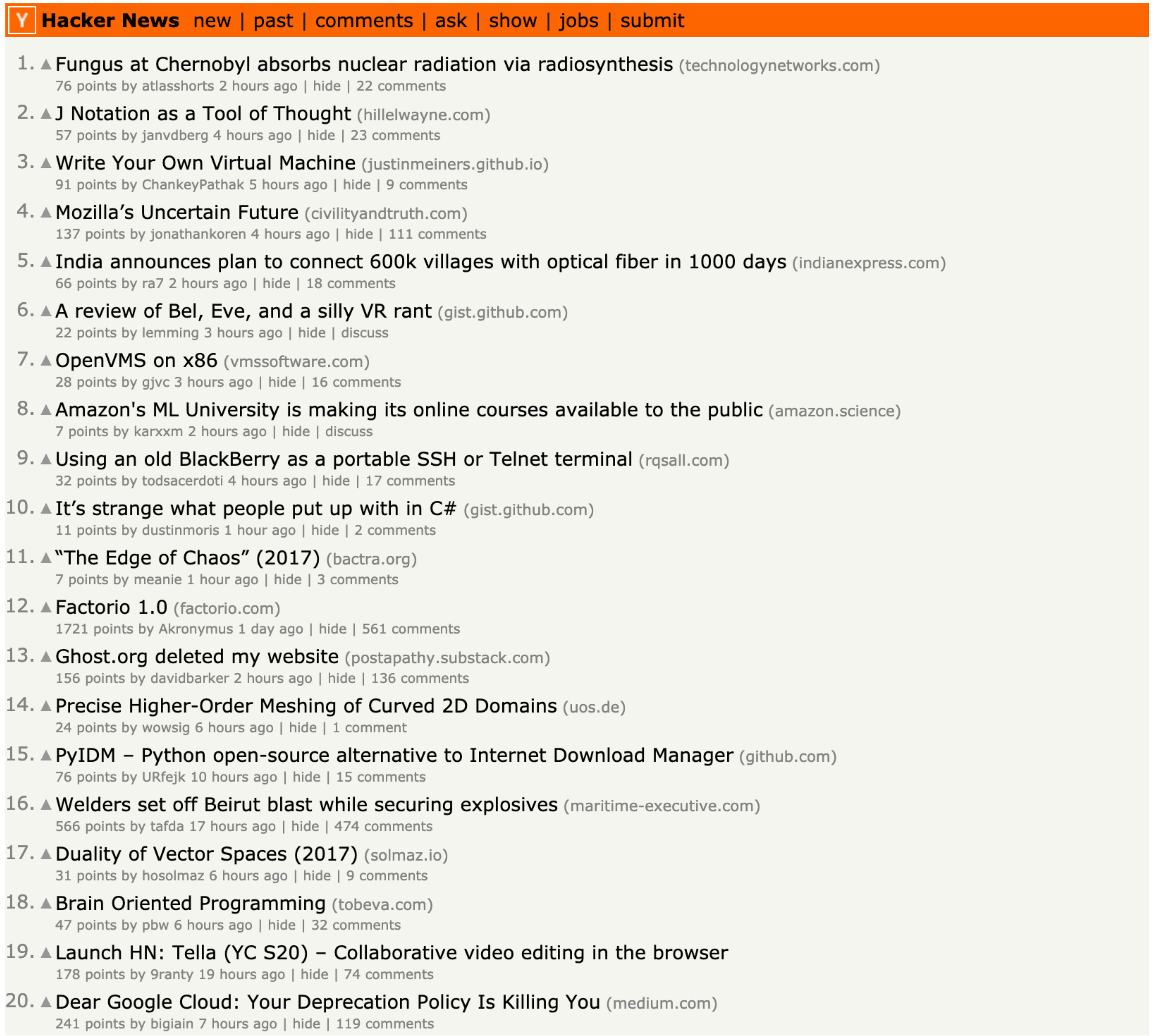
```
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 %in% c("Uganda", "India", "Canada", "Mexico")) %>%
  group_by(item1) %>%
  top_n(16, abs(correlation)) %>%
  mutate(item2 = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(correlation, item2)) +
  geom_col() +
  facet_wrap(~ item1, scales = "free_y") +
  scale_y_reordered() +
  labs(x = "Correlation", y = "")
```

ggplot2



Pairwise example:
Word co-occurrence

Hacker News titles


Y **Hacker News** new | past | comments | ask | show | jobs | submit

1. ▲ Fungus at Chernobyl absorbs nuclear radiation via radiosynthesis (technologynetworks.com)
76 points by atlasshorts 2 hours ago | hide | 22 comments
2. ▲ J Notation as a Tool of Thought (hillelwayne.com)
57 points by janvdberg 4 hours ago | hide | 23 comments
3. ▲ Write Your Own Virtual Machine (justinmeiners.github.io)
91 points by ChankeyPathak 5 hours ago | hide | 9 comments
4. ▲ Mozilla's Uncertain Future (civilityandtruth.com)
137 points by jonathankoren 4 hours ago | hide | 111 comments
5. ▲ India announces plan to connect 600k villages with optical fiber in 1000 days (indianexpress.com)
66 points by ra7 2 hours ago | hide | 18 comments
6. ▲ A review of Bel, Eve, and a silly VR rant (gist.github.com)
22 points by lemming 3 hours ago | hide | discuss
7. ▲ OpenVMS on x86 (vmssoftware.com)
28 points by gjvc 3 hours ago | hide | 16 comments
8. ▲ Amazon's ML University is making its online courses available to the public (amazon.science)
7 points by karxxm 2 hours ago | hide | discuss
9. ▲ Using an old BlackBerry as a portable SSH or Telnet terminal (rqsall.com)
32 points by todsacerdoti 4 hours ago | hide | 17 comments
10. ▲ It's strange what people put up with in C# (gist.github.com)
11 points by dustinmoris 1 hour ago | hide | 2 comments
11. ▲ "The Edge of Chaos" (2017) (bactra.org)
7 points by meanie 1 hour ago | hide | 3 comments
12. ▲ Factorio 1.0 (factorio.com)
1721 points by Akronymus 1 day ago | hide | 561 comments
13. ▲ Ghost.org deleted my website (postapathy.substack.com)
156 points by davidbarker 2 hours ago | hide | 136 comments
14. ▲ Precise Higher-Order Meshing of Curved 2D Domains (uos.de)
24 points by wowsig 6 hours ago | hide | 1 comment
15. ▲ PyIDM – Python open-source alternative to Internet Download Manager (github.com)
76 points by URfejk 10 hours ago | hide | 15 comments
16. ▲ Welders set off Beirut blast while securing explosives (maritime-executive.com)
566 points by tafda 17 hours ago | hide | 474 comments
17. ▲ Duality of Vector Spaces (2017) (solmaz.io)
31 points by hosolmaz 6 hours ago | hide | 9 comments
18. ▲ Brain Oriented Programming (tobeva.com)
47 points by pbw 6 hours ago | hide | 32 comments
19. ▲ Launch HN: Tella (YC S20) – Collaborative video editing in the browser
178 points by 9ranty 19 hours ago | hide | 74 comments
20. ▲ Dear Google Cloud: Your Deprecation Policy Is Killing You (medium.com)
241 points by bigiain 7 hours ago | hide | 119 comments

```
# A tibble: 99,996 x 3
   post_id date       title
     <int> <date>     <chr>
 1       1 2019-01-01 Learn the Rules Like a Pro, So You Can …
 2       2 2019-01-01 Upgrading the Nginx Executable on the F…
 3       3 2019-01-01 Trendism and cognitive stagnation
 4       4 2019-01-01 DNS Records Checker
 5       5 2019-01-01 UX Designer's guide to effective retros…
 6       6 2019-01-01 Nevralgiile faciale tratamente naturiste
 7       7 2019-01-01 Online tutoring app Byju touches $3.8B …
 8       8 2019-01-01 How to Play PUBG on Pc Using This Simpl…
 9       9 2019-01-01 Simya Koleji Türkiye Geneli Bursluluk S…
10      10 2019-01-01 At the twilight of Moore’s Law
# … with 99,986 more rows
```

Adapted from Training, Evaluating, and Interpreting Topic Models by Julia Silge

Tokenizing Hacker News titles with tidytext

```r
hacker_news_words <- hacker_news_text %>%
  unnest_tokens(word, title) %>%
  anti_join(stop_words, by = "word") %>%
  filter(!str_detect(word, "[0-9]+")) %>%
  add_count(word, name = "word_total") %>%
  filter(word_total >= 250)
```

```
# A tibble: 120,106 x 3
   post_id date       word
     <int> <date>     <chr>
 1       1 2019-01-01 learn
 2       1 2019-01-01 pro
 3       5 2019-01-01 guide
 4       7 2019-01-01 online
 5       7 2019-01-01 app
 6       8 2019-01-01 play
 7       8 2019-01-01 simple
 8      10 2019-01-01 law
 9      15 2019-01-01 data
10      16 2019-01-01 design
# … with 120,096 more rows
```

Pairwise co-occurrences of words

```r
hacker_news_words %>%
  pairwise_cor(word, post_id, sort = TRUE)
```

$$\phi = \frac{n_{11}n_{00} - n_{10}n_{01}}{\sqrt{n_{1\bullet}n_{0\bullet}n_{\bullet 0}n_{\bullet 1}}}$$

Phi coefficient

```
# A tibble: 51,302 x 3
   item1    item2    correlation
   <chr>    <chr>          <dbl>
 1 machine  learning       0.505
 2 learning machine        0.505
 3 media    social         0.493
 4 social   media          0.493
 5 networks neural         0.472
 6 neural   networks       0.472
 7 climate  change         0.443
 8 change   climate        0.443
 9 react    native         0.356
10 native   react          0.356
# … with 51,292 more rows
```

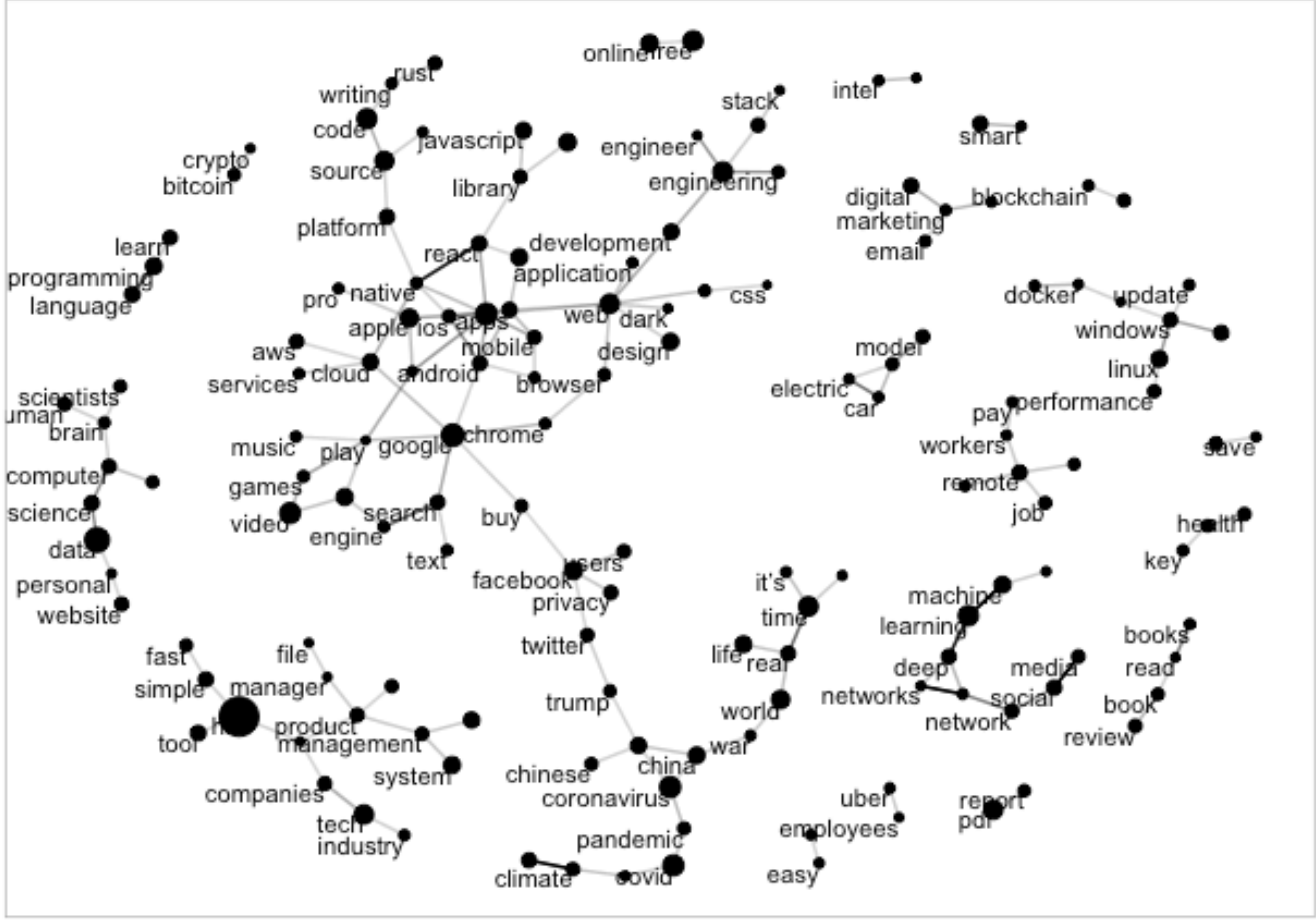
Pairwise co-occurrences of words

```r
hacker_news_words %>%
  pairwise_cor(word, post_id, sort = TRUE) %>%
  filter(item1 == "data")
```

```
# A tibble: 226 x 3
   item1 item2       correlation
   <chr> <chr>             <dbl>
 1 data  science          0.140
 2 data  personal         0.0377
 3 data  scientists       0.0351
 4 data  user             0.0329
 5 data  access           0.0294
 6 data  analysis         0.0291
 7 data  privacy          0.0264
 8 data  machine          0.0177
 9 data  cloud            0.0140
10 data  learning         0.0138
# … with 216 more rows
```

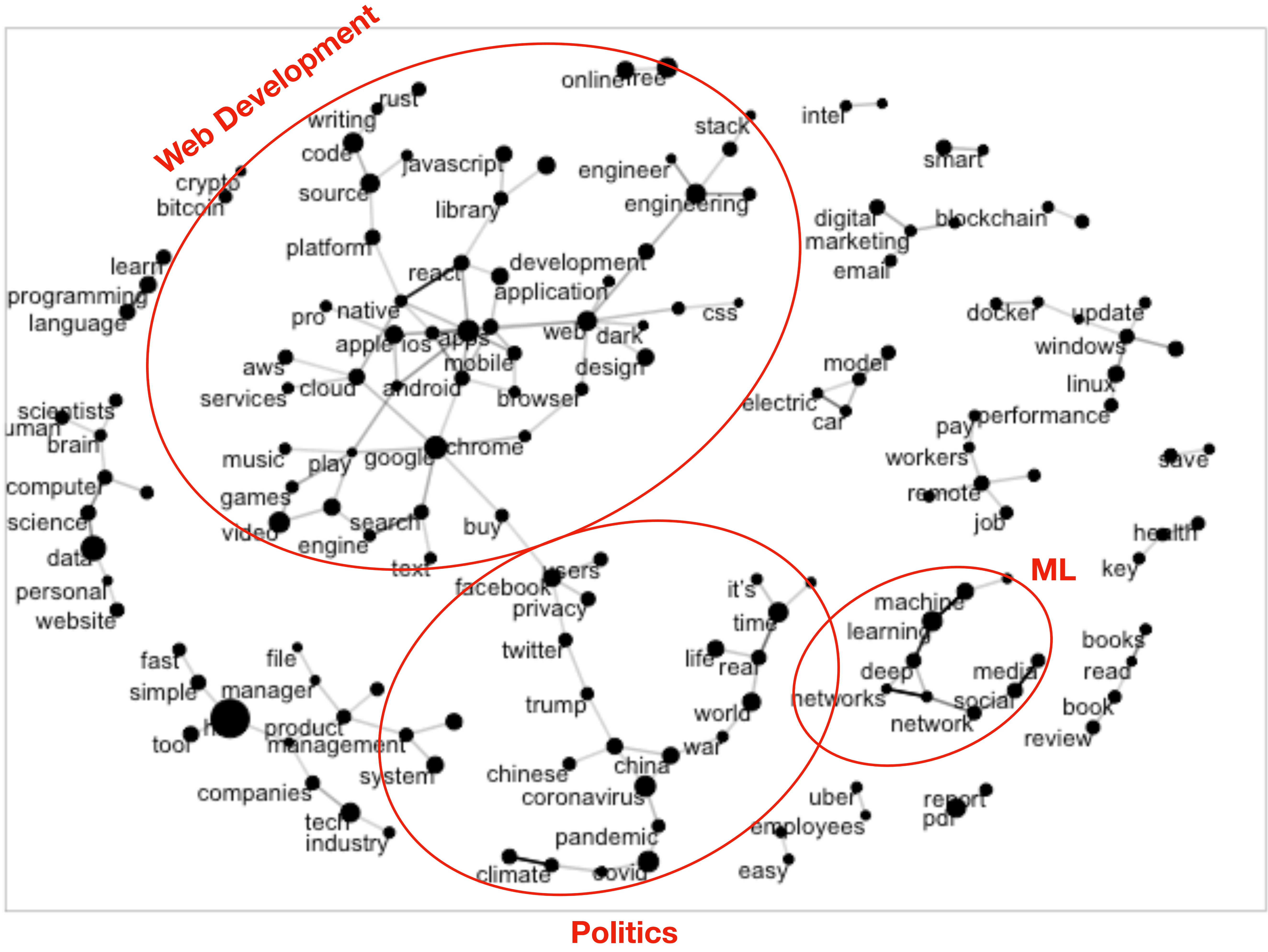
Network plots with tidy graph + ggraph

```r
library(ggraph)
library(tidygraph)

word_counts <- hacker_news_words %>%
  count(word, sort = TRUE)

hacker_news_words %>%
  pairwise_cor(word, post_id, sort = TRUE) %>%
  head(300) %>%
  as_tbl_graph() %>%
  inner_join(word_counts, by = c(name = "word")) %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation)) +
  geom_node_point(aes(size = n)) +
  geom_node_text(aes(label = name), check_overlap = TRUE,
                 vjust = 1, hjust = 1, size = 3) +
  theme(legend.position = "none")
```



Network plots with tidy graph + ggraph

```r
library(ggraph)
library(tidygraph)

word_counts <- hacker_news_words %>%
  count(word, sort = TRUE)

hacker_news_words %>%
  pairwise_cor(word, post_id, sort = TRUE) %>%
  head(300) %>%
  as_tbl_graph() %>%
  inner_join(word_counts, by = c(name = "word")) %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation)) +
  geom_node_point(aes(size = n)) +
  geom_node_text(aes(label = name), check_overlap = TRUE,
                 vjust = 1, hjust = 1, size = 3) +
  theme(legend.position = "none")
```

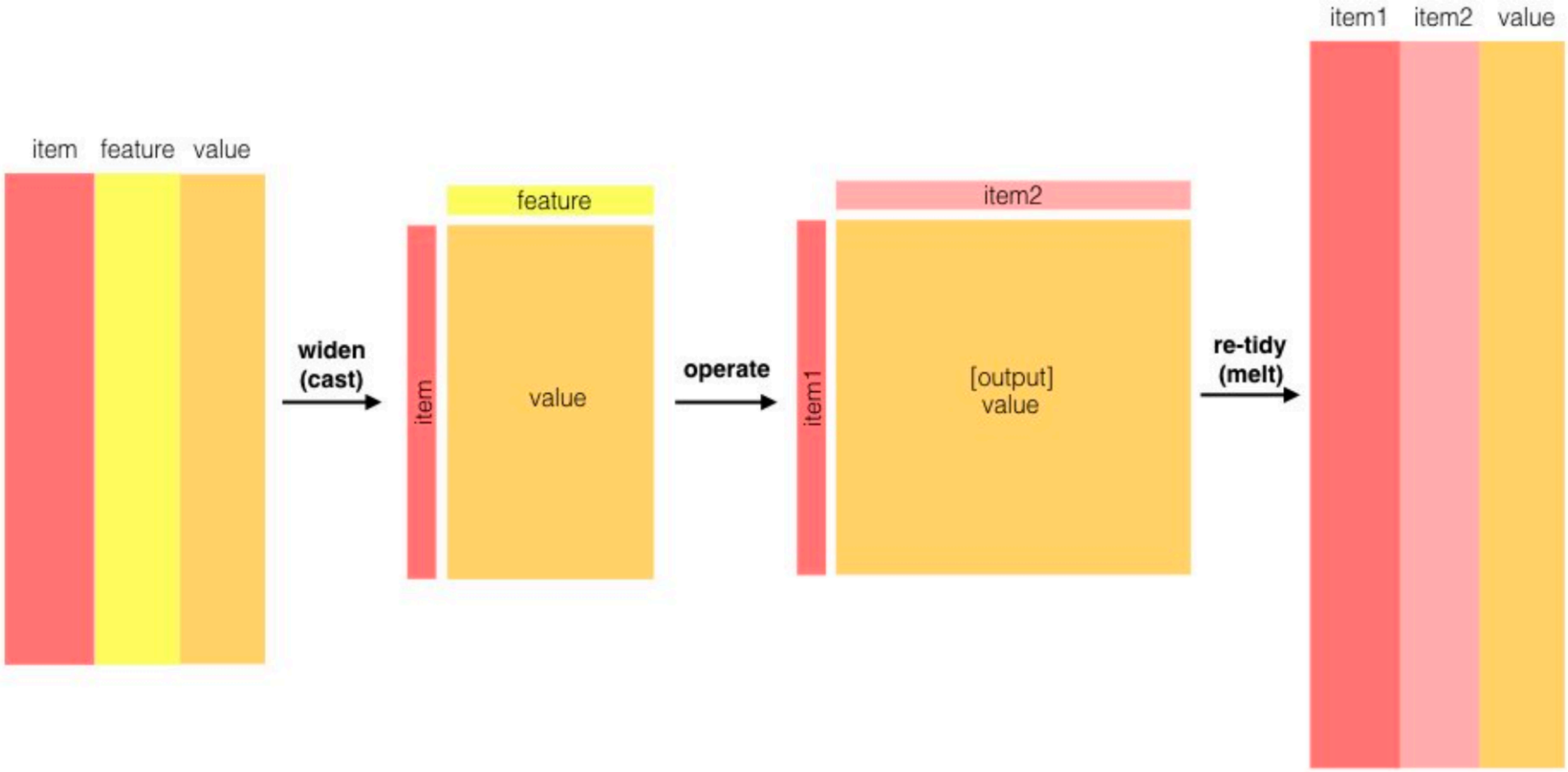


Other pairwise operations in widyr

- `pairwise_count` How often do these two items appear together?
- `pairwise_dist` Euclidean/Manhattan/etc distance
- `pairwise_similarity` Cosine similarity
- `pairwise_pmi` Pairwise mutual information
- `pairwise_delta` Calculate Burrows delta (for authorship attribution)

Widely example:
clustering + dimensionality reduction

The widen-operate-retidy pattern is very flexible

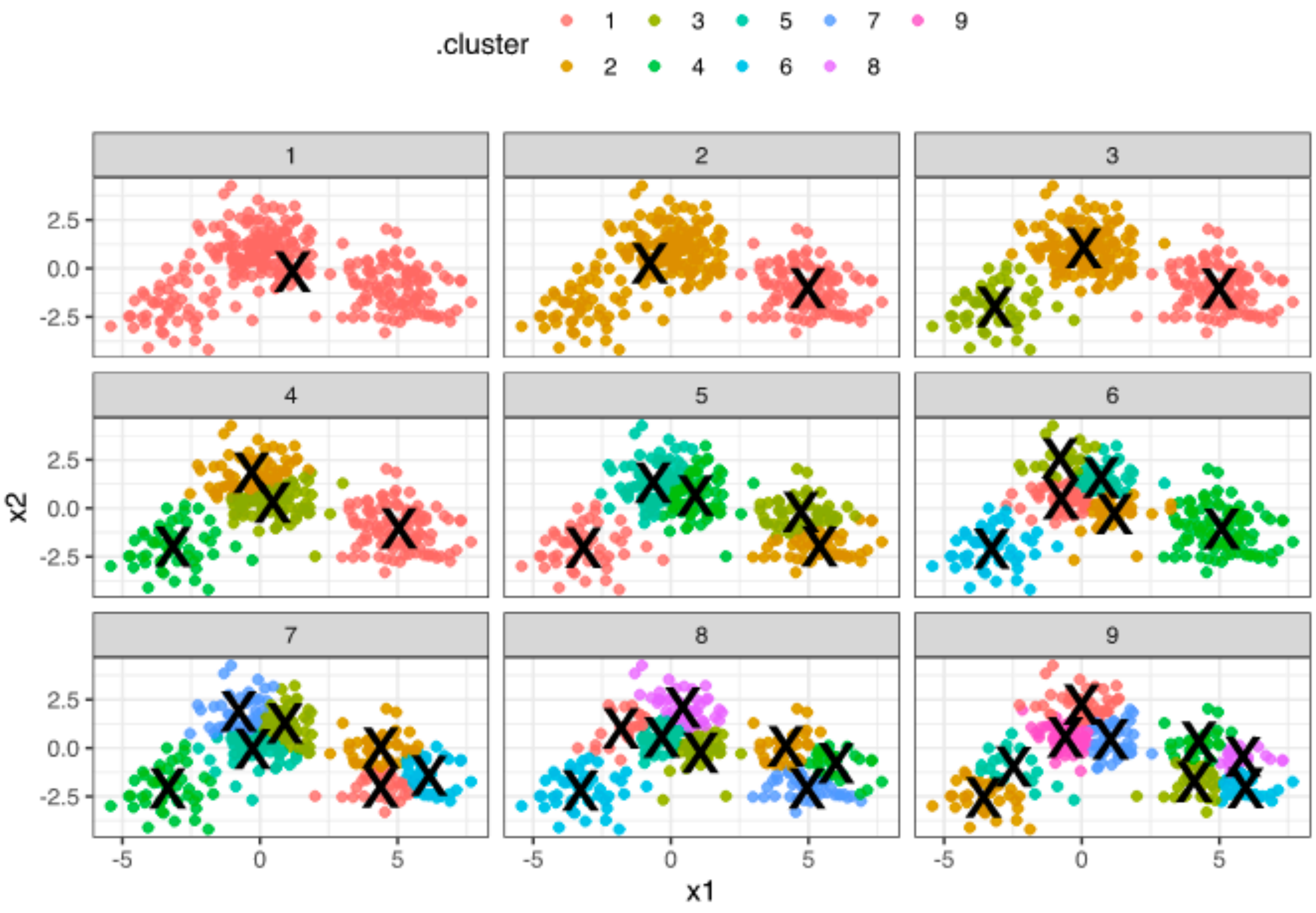


Input is tidy

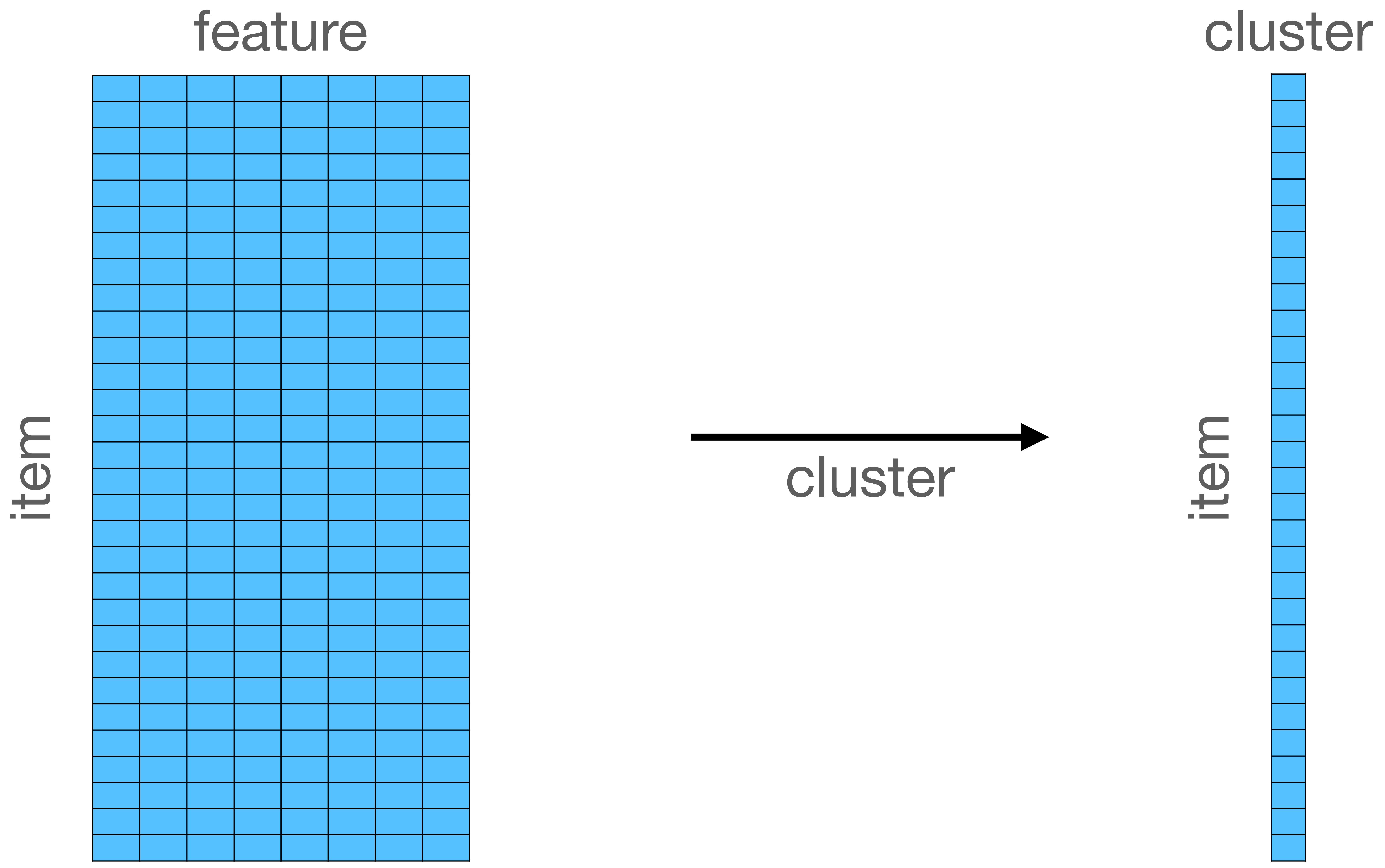
Middle is “wide” matrices

Output is tidy

K-means is a classic approach to clustering



Clustering is an example of a “wide” operation



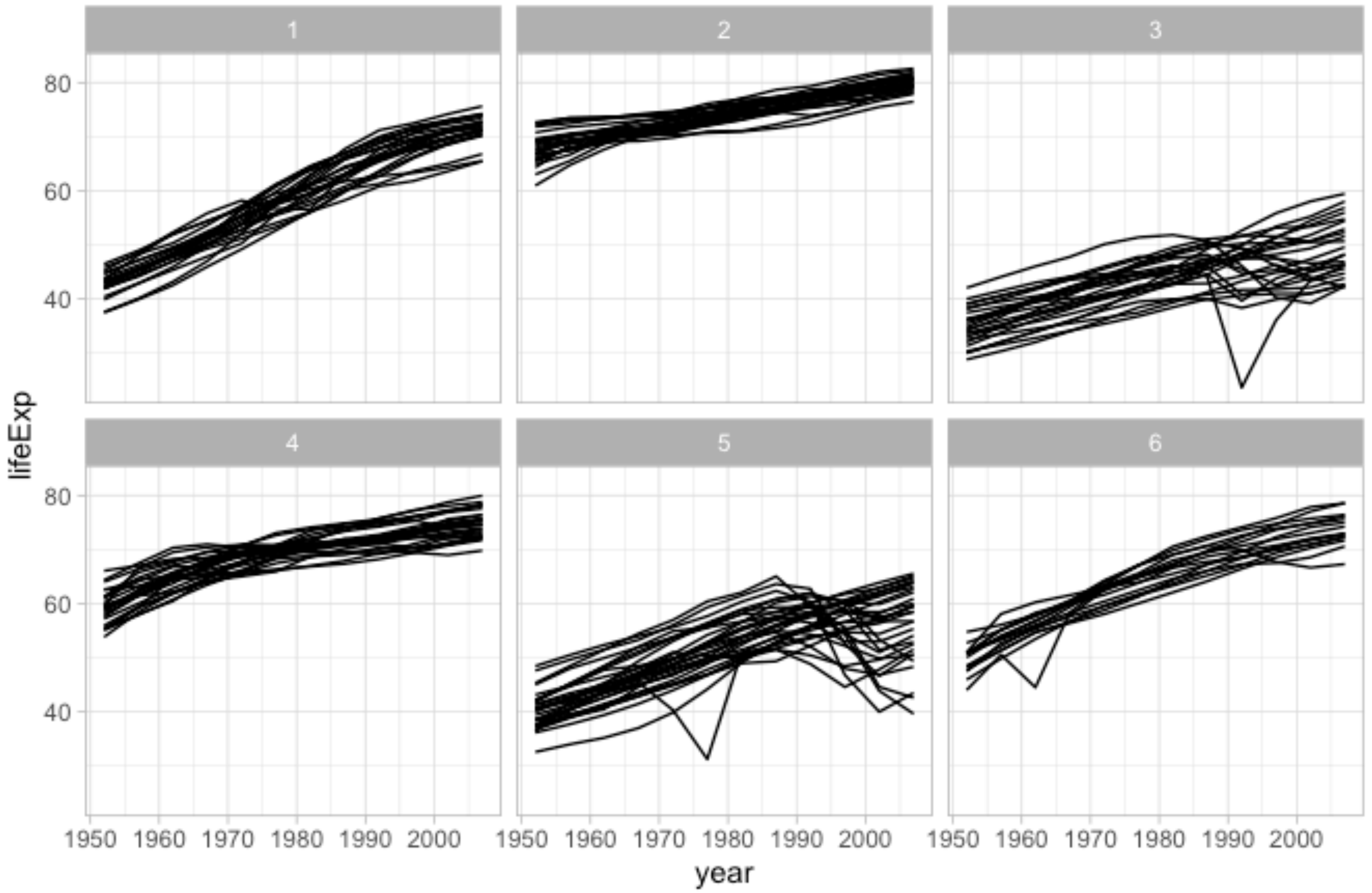
`widely_kmeans` performs clustering on tidy data

```r
gapminder %>%
  widely_kmeans(country, year, lifeExp, k = 6)
```

```
# A tibble: 142 x 2
   country     cluster
   <fct>       <fct>
 1 Algeria     1
 2 Egypt       1
 3 El Salvador 1
 4 Guatemala   1
 5 Honduras    1
 6 Indonesia   1
 7 Iran        1
 8 Jordan      1
 9 Libya       1
10 Mongolia    1
# … with 132 more rows
```

`widely_kmeans` performs clustering on tidy data

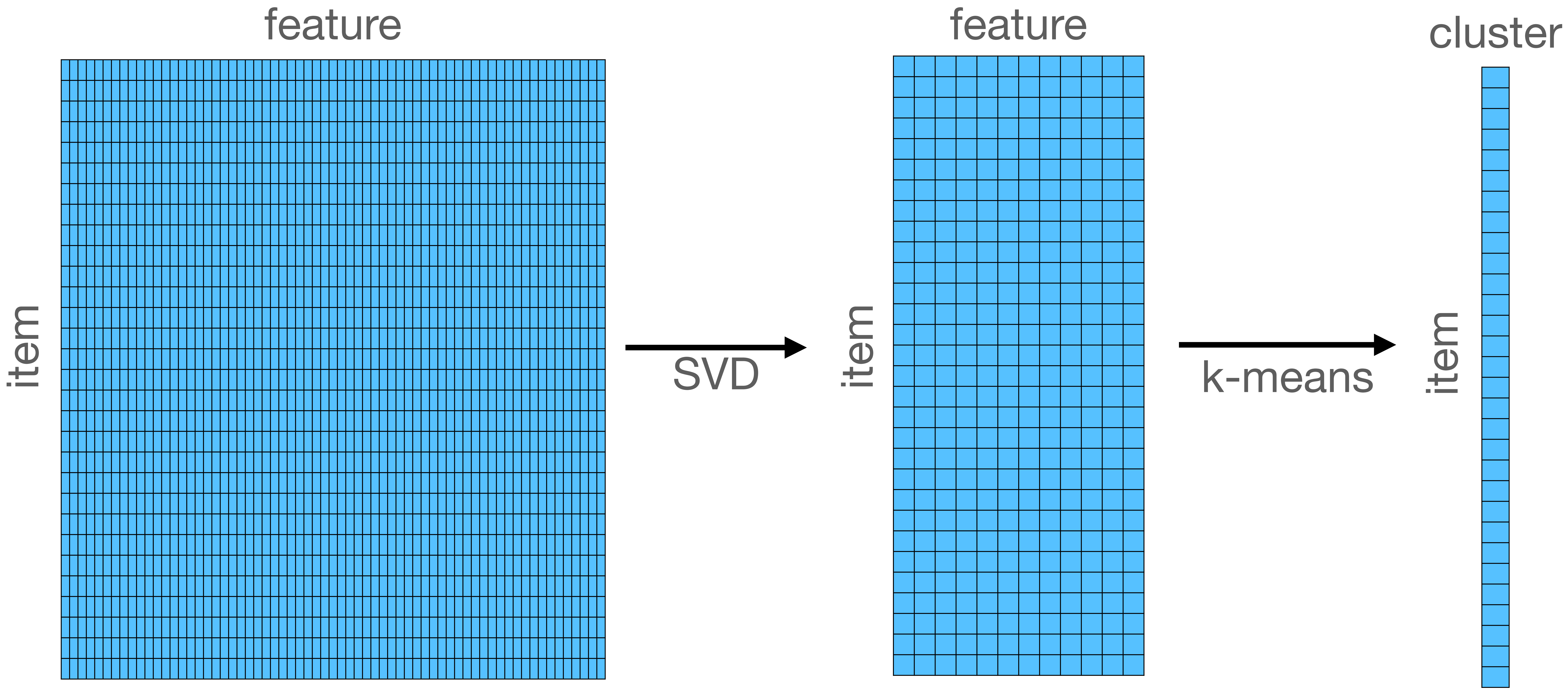
```r
clusters <- gapminder %>%
  widely_kmeans(country, year, lifeExp, k = 6)

gapminder %>%
  inner_join(clusters, by = "country") %>%
  ggplot(aes(year, lifeExp, group = country)) +
  geom_line() +
  facet_wrap(~ cluster)
```



widyr (development) offers three `widely_` functions

- `widely_kmeans` K-means clustering
- `widely_hclust` Hierarchical clustering on distances
- `widely_svd` Singular value decomposition for dimensionality reduction

Dimensionality reduction + clustering



Dimensionality reduction + clustering

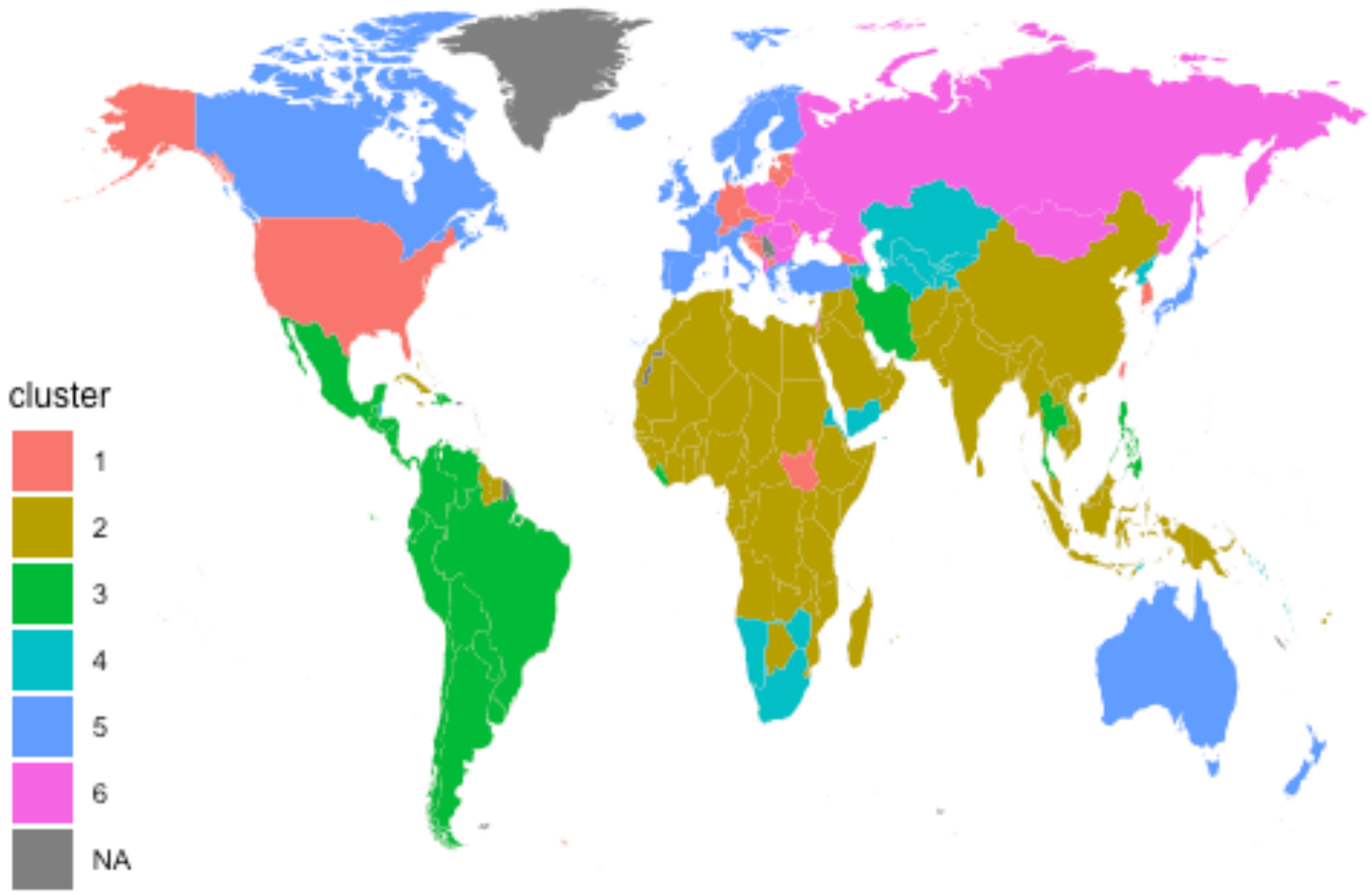
```
# A tibble: 733,404 x 4
    rcid country                  country_code  vote
   <int> <chr>                    <chr>        <dbl>
 1     3 United States of America US               1
 2     3 Canada                   CA              -1
 3     3 Cuba                     CU               1
 4     3 Haiti                    HT               1
 5     3 Dominican Republic       DO               1
 6     3 Mexico                   MX               1
 7     3 Guatemala                GT               1
 8     3 Honduras                 HN               1
 9     3 El Salvador              SV               1
10     3 Nicaragua                NI               1
# … with 733,394 more rows
```

```r
votes %>%
  widely_svd(country, rcid, vote, nv = 16) %>%
  widely_kmeans(country, dimension, value, k = 6)
```

```
# A tibble: 197 x 2
   country           cluster
   <chr>             <fct>
 1 Algeria           1
 2 Bahrain           1
 3 Barbados          1
 4 Bhutan            1
 5 Botswana          1
 6 Burundi           1
 7 China             1
 8 Equatorial Guinea 1
 9 Fiji              1
10 Gambia            1
# … with 187 more rows
```

Describing voting blocs through clustering

```r
library(maps)
library(fuzzyjoin)

map_clusters <- votes %>%
  widely_svd(country_code, rcid, vote, nv = 24) %>%
  widely_kmeans(country_code, dimension, value, k = 6) %>%
  inner_join(iso3166, by = c(country_code = "a2"))

map_data("world") %>%
  filter(region != "Antarctica") %>%
  regex_left_join(map_clusters, by = c("region" = "mapname")) %>%
  ggplot(aes(long, lat, group = group, fill = cluster)) +
  geom_polygon() +
  ggthemes::theme_map()
```

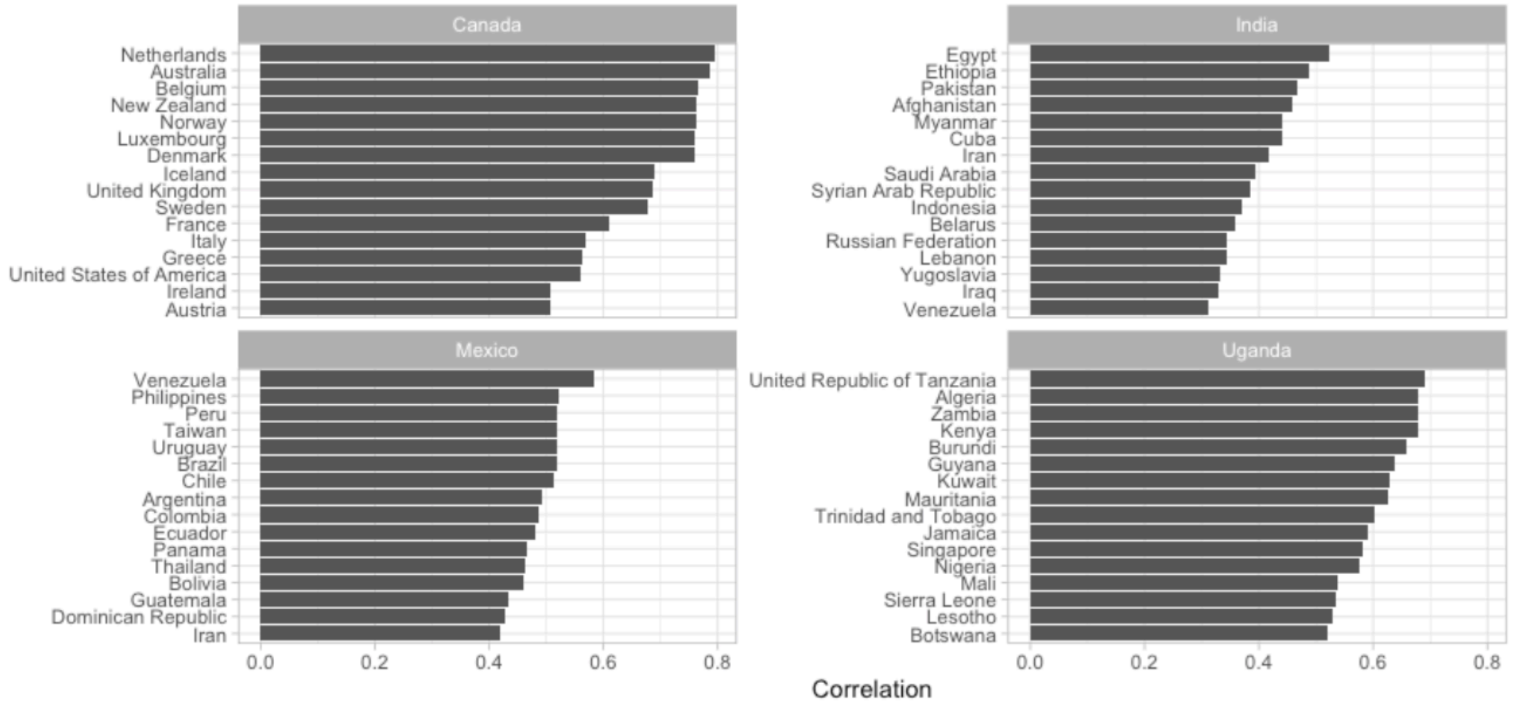


Conclusion

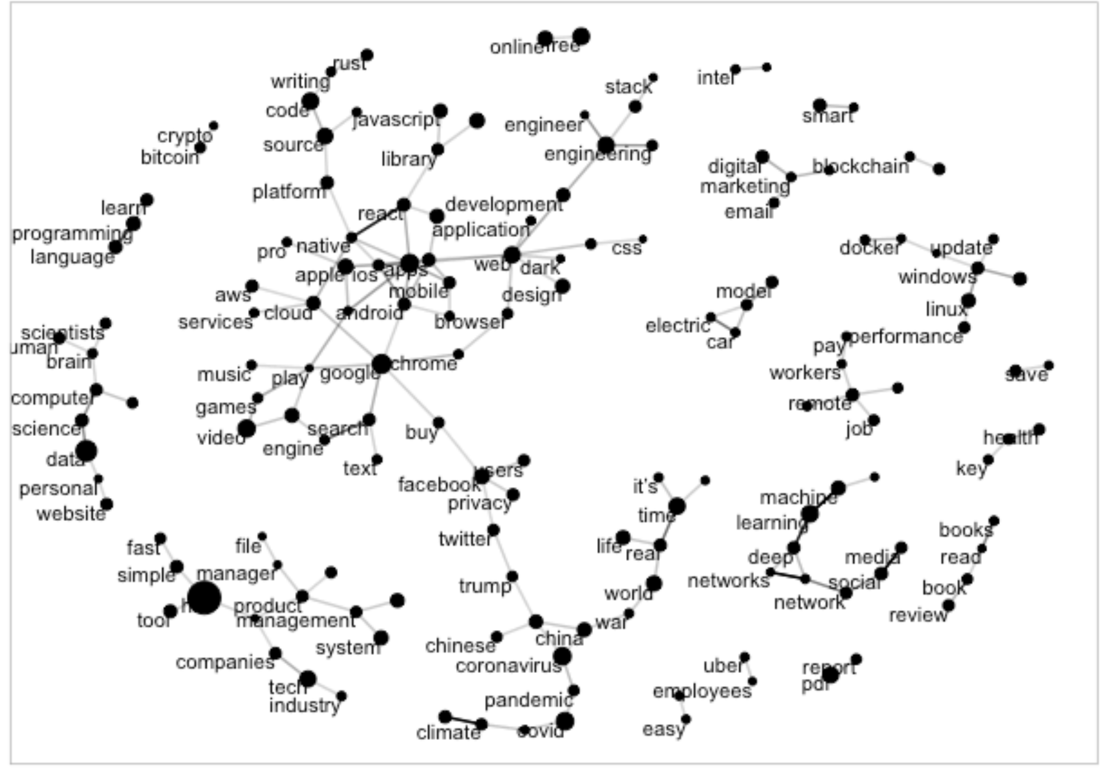
“No matter how complex and polished the individual operations are, it is often the quality of the glue that most directly determines the power of the system.”

-Hal Abelson

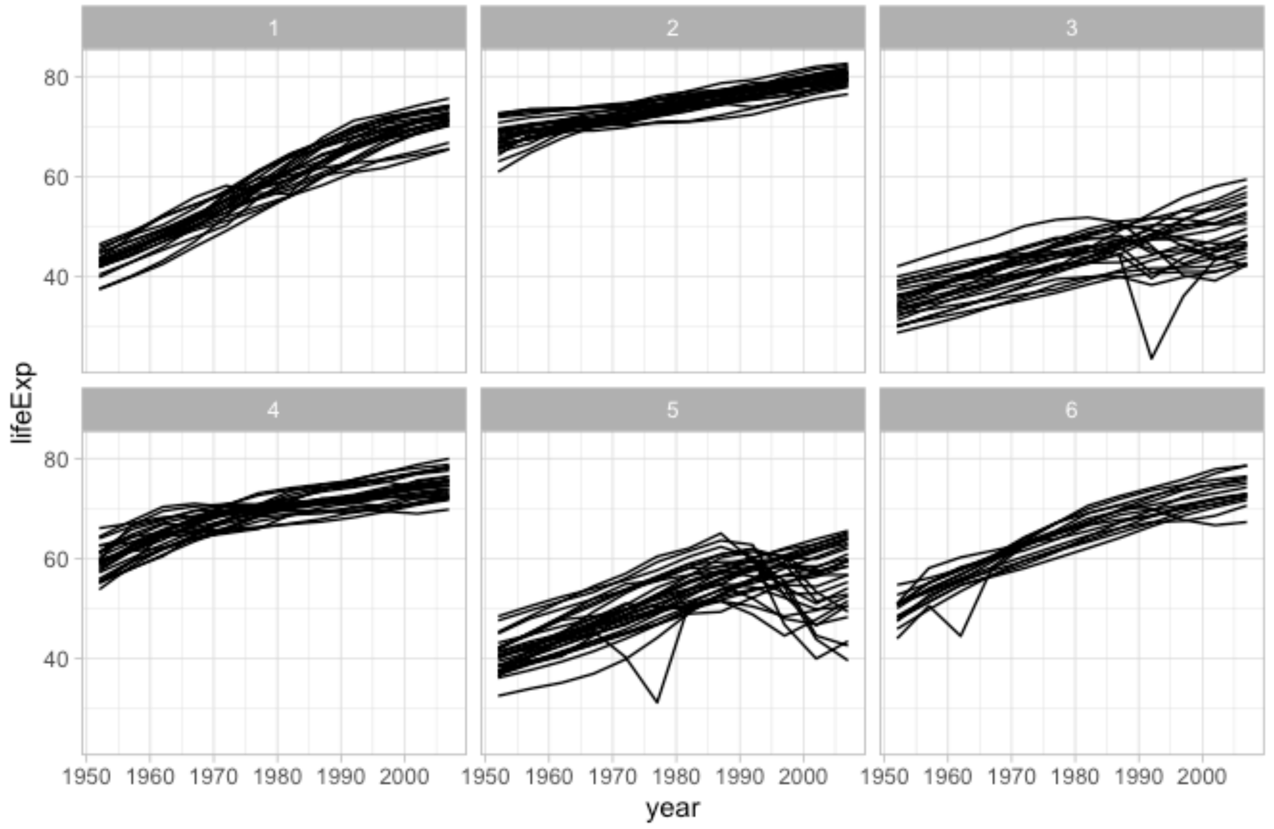
```r
votes %>%
  pairwise_cor(country, rcid, vote, sort = TRUE) %>%
  filter(item1 %in% c("Uganda", "India", "Canada", "Mexico")) %>%
  group_by(item1) %>%
  top_n(16, abs(correlation)) %>%
  mutate(item2 = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(correlation, item2)) +
  geom_col() +
  facet_wrap(~ item1, scales = "free_y") +
  scale_y_reordered() +
  labs(x = "Correlation", y = "")
```



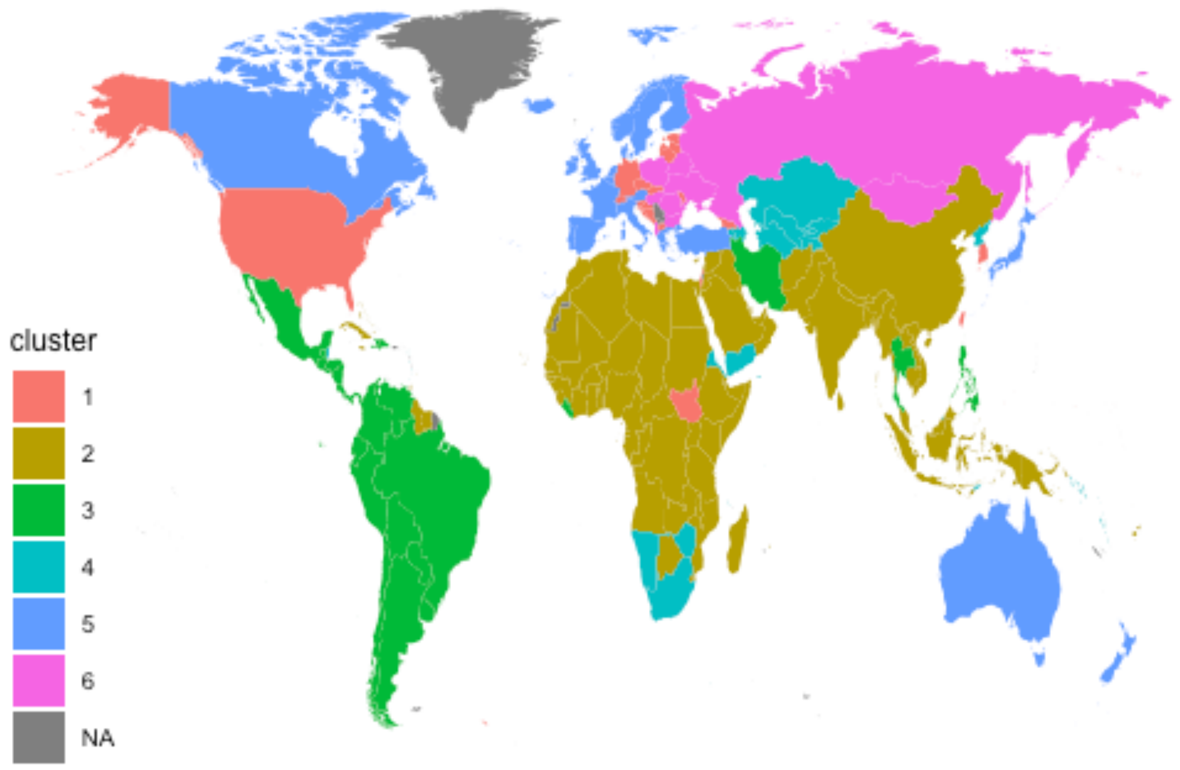

```r
library(ggraph)
library(tidygraph)

word_counts <- hacker_news_words %>%
  count(word, sort = TRUE)

hacker_news_words %>%
  pairwise_cor(word, post_id, sort = TRUE) %>%
  head(300) %>%
  as_tbl_graph() %>%
  inner_join(word_counts, by = c(name = "word")) %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation)) +
  geom_node_point(aes(size = n)) +
  geom_node_text(aes(label = name), check_overlap = TRUE,
                 vjust = 1, hjust = 1, size = 3) +
  theme(legend.position = "none")
```




```r
clusters <- gapminder %>%
  widely_kmeans(country, year, lifeExp, k = 6)

gapminder %>%
  inner_join(clusters, by = "country") %>%
  ggplot(aes(year, lifeExp, group = country)) +
  geom_line() +
  facet_wrap(~ cluster)
```




```r
library(maps)
library(fuzzyjoin)

map_clusters <- votes %>%
  widely_svd(country_code, rcid, vote, nv = 24) %>%
  widely_kmeans(country_code, dimension, value, k = 6) %>%
  inner_join(iso3166, by = c(country_code = "a2"))

map_data("world") %>%
  filter(region != "Antarctica") %>%
  regex_left_join(map_clusters, by = c("region" = "mapname")) %>%
  ggplot(aes(long, lat, group = group, fill = cluster)) +
  geom_polygon() +
  ggthemes::theme_map()
```



Once “wide” operations are atomic actions, you can do a lot with a little code

Thank you

- Lander Analytics
  - Jared Lander
  - Amada Echeverria

Heap

@drob

www.varianceexplained.org

VARIANCE EXPLAINED

ABOUT ME POSTS LEARN R TEXT MINING IN R INTRODUCTION TO EMPIRICAL BAYES

David Robinson

Chief Data Scientist at DataCamp, works in R and Python.

- Email
- Twitter
- Github
- Stack Overflow

Subscribe

Your email

This is the homepage and blog of David Robinson, Chief Data Scientist at DataCamp. For more about me, see here.

Recent Posts

The 'knight on an infinite chessboard' puzzle: efficient simulation in R *December 10, 2018*
A simulation of a probabilistic puzzle from the Riddler column on FiveThirtyEight.

Exploring college major and income: a live data analysis in R *October 16, 2018*
A live screencast of an exploratory data analysis from the Tidy Tuesday series. This one explores college major and income data from 538.

Who wrote the anti-Trump New York Times op-ed? Using tidytext to find document similarity *September 06, 2018*
An analysis of an anonymous op-ed in the New York Times, using document similarity metrics to match it to Twitter accounts.

Scientific debt *May 10, 2018*
Introducing an analogy to 'technical debt' for data scientists.