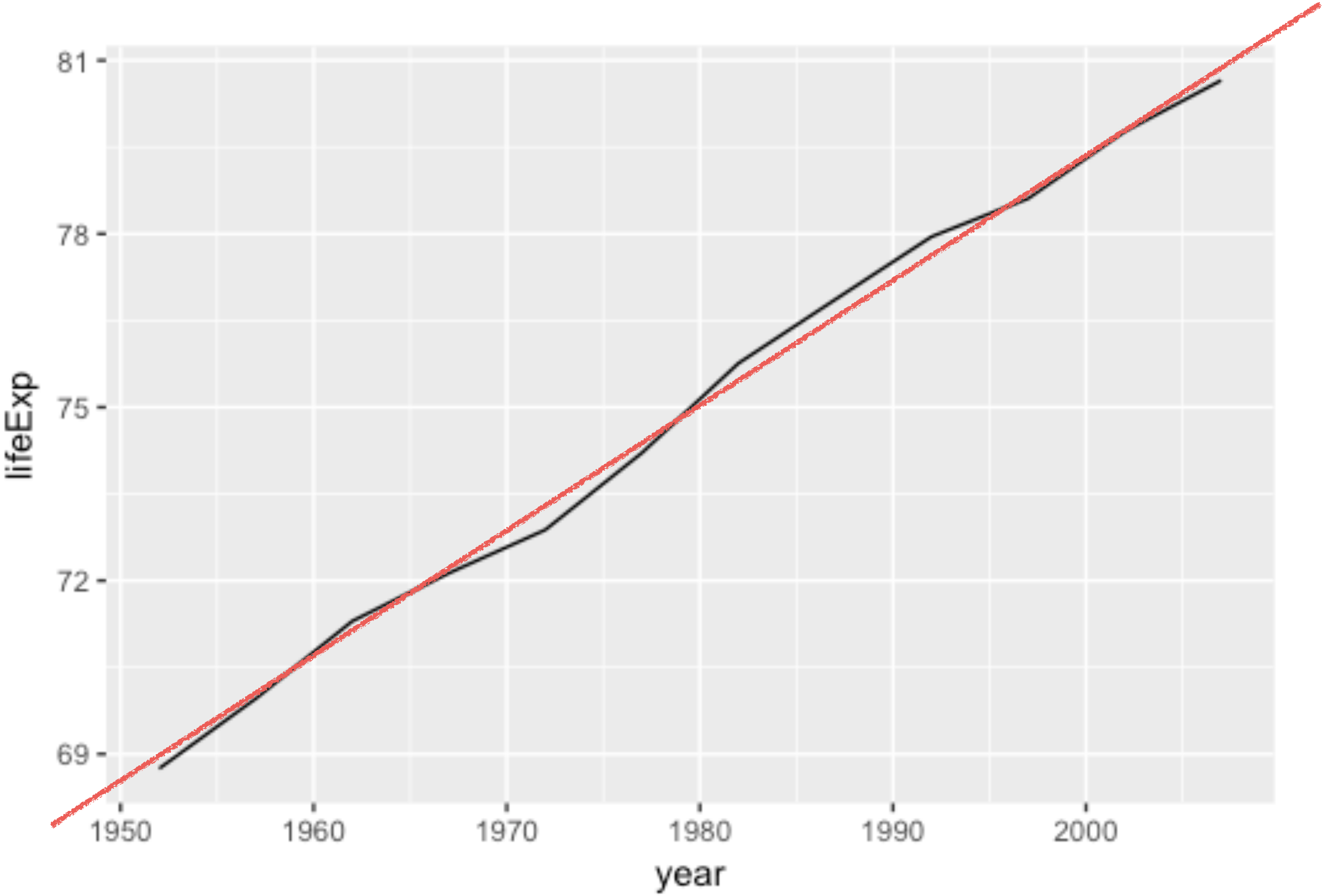
Modelling:
the basics

Your Turn 1

Take another look at life expectancy over time for Canada.

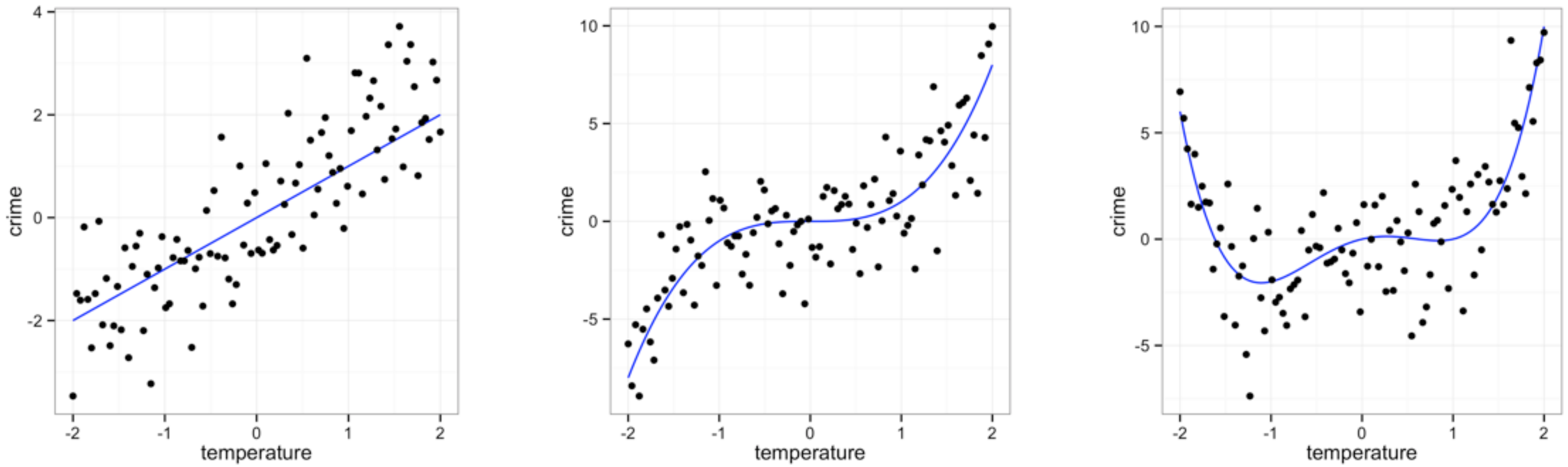
What do you notice?



Adapted from 'Master the tidyverse' CC by RStudio

Models

A low dimensional description of a higher dimensional data set.



modelr

(Popular) modeling functions in R

function	package	fits
lm()	stats	linear models
glm()	stats	generalized linear models
gam()	mgcv	generalized additive models
glmnet()	glmnet	penalized linear models
rlm()	MASS	robust linear models
rpart()	rpart	trees
randomForest()	randomForest	random forests
xgboost()	xgboost	gradient boosting machines

(Popular) modeling functions in R

function	package	fits
lm()	**stats**	**linear models**
glm()	stats	generalized linear models
gam()	mgcv	generalized additive models
glmnet()	glmnet	penalized linear models
rlm()	MASS	robust linear models
rpart()	rpart	trees
randomForest()	randomForest	random forests
xgboost()	xgboost	gradient boosting machines

lm()

lm()

Fit a linear model to data

```
lm(lifeExp ~ year, data = canada)
```

A formula that describes the model equation

The data set

Adapted from 'Master the tidyverse' CC by RStudio

formulas

Formula only needs to include the response and predictors

$$y = \alpha + \beta x + \epsilon$$

```
y ~ x
```

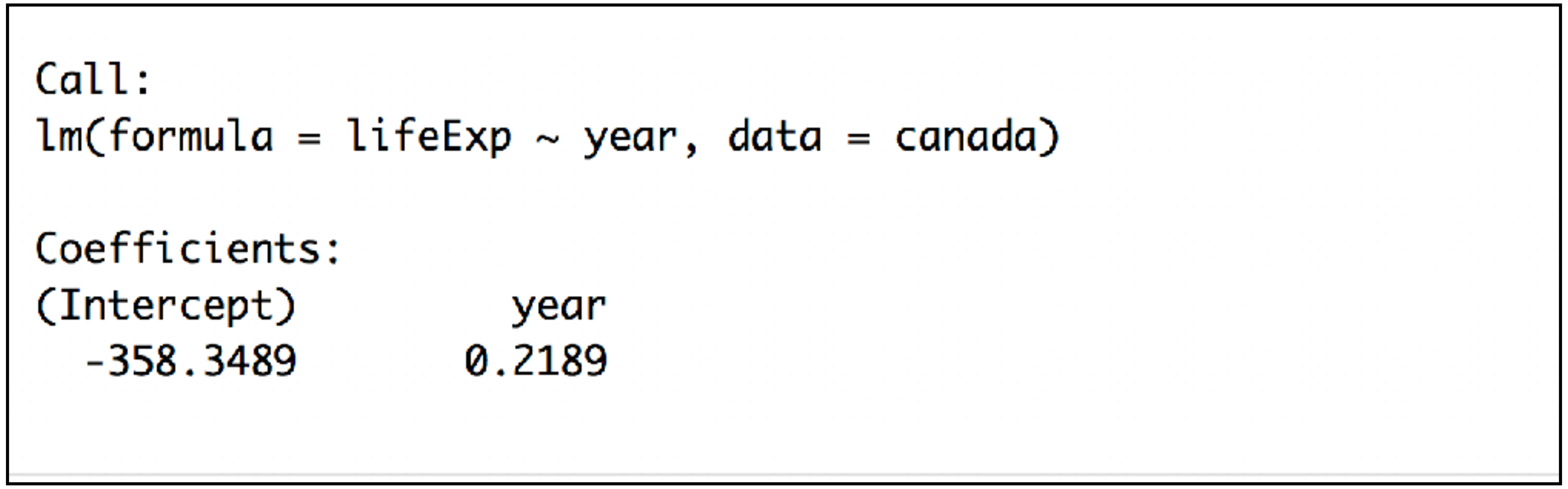
Your Turn 2

Run the chunk to fit the model.

What kind of output do you get?

Explore `canada_lm` by clicking on it in the environment, what's inside the object?

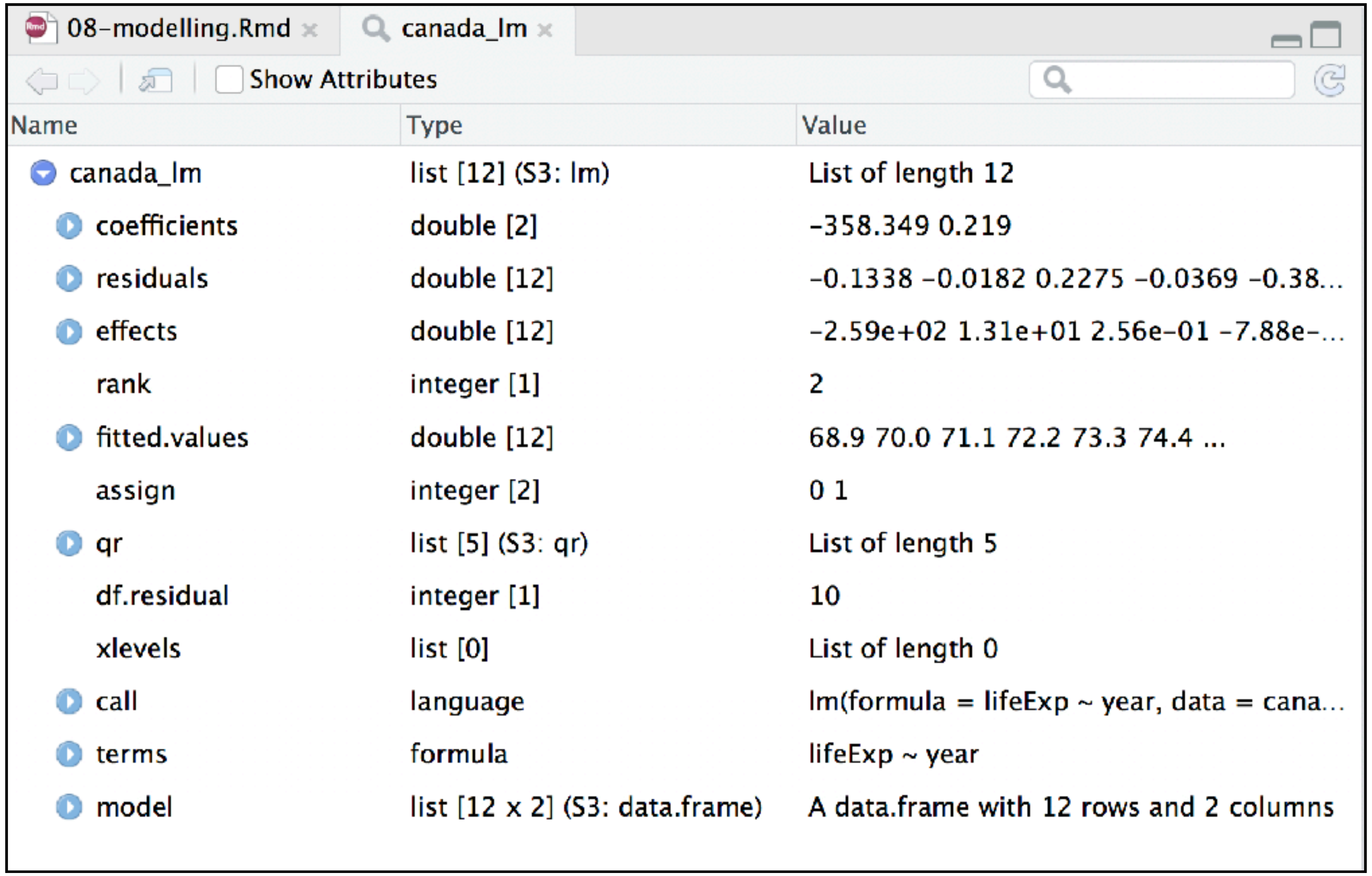
04:00



```
Call:
lm(formula = lifeExp ~ year, data = canada)

Coefficients:
(Intercept)         year
  -358.3489       0.2189
```

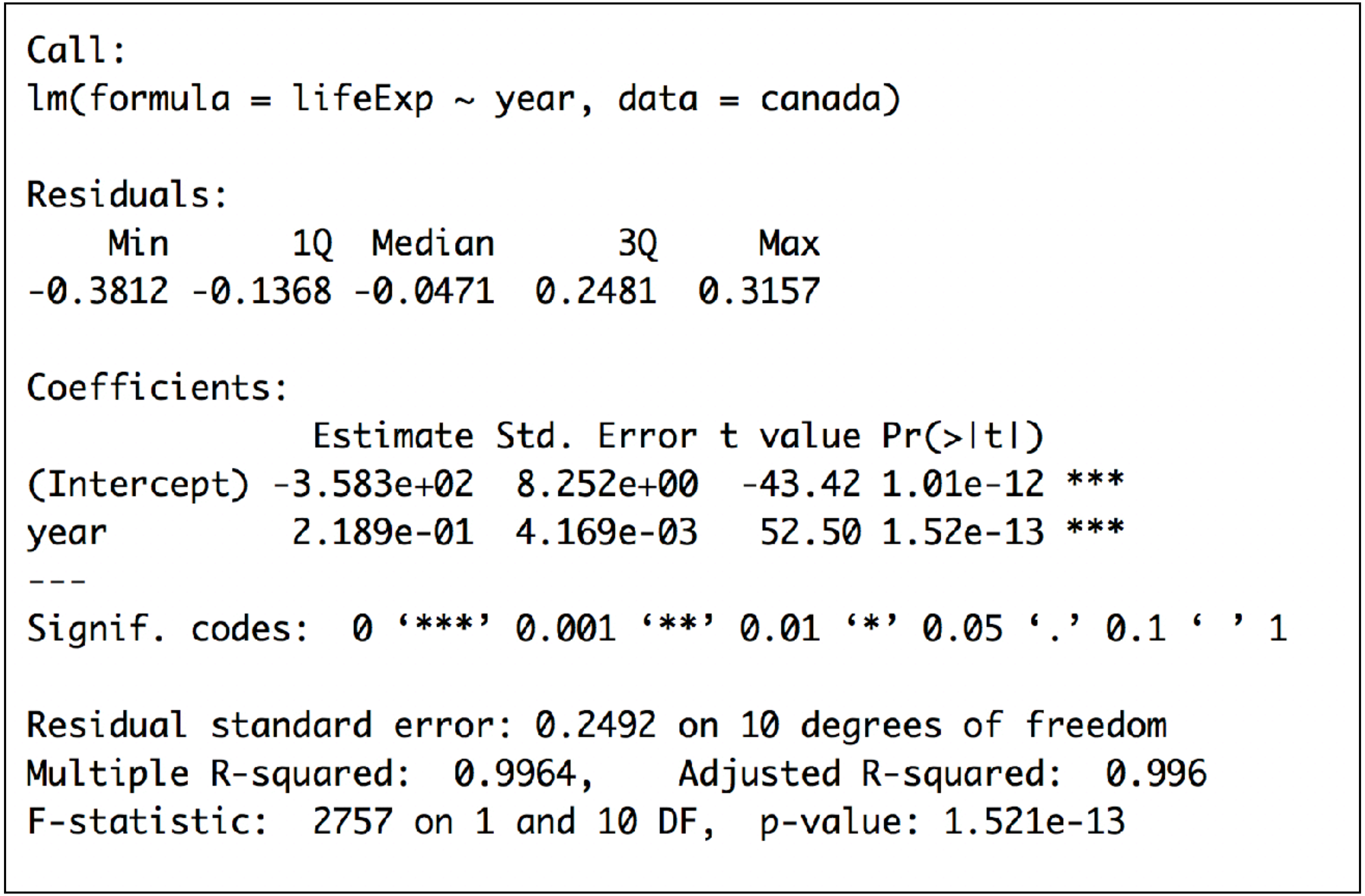
Asking for the object prints a minimal summary



Name	Type	Value
canada_lm	list [12] (S3: lm)	List of length 12
coefficients	double [2]	-358.349 0.219
residuals	double [12]	-0.1338 -0.0182 0.2275 -0.0369 -0.38...
effects	double [12]	-2.59e+02 1.31e+01 2.56e-01 -7.88e-...
rank	integer [1]	2
fitted.values	double [12]	68.9 70.0 71.1 72.2 73.3 74.4 ...
assign	integer [2]	0 1
qr	list [5] (S3: qr)	List of length 5
df.residual	integer [1]	10
xlevels	list [0]	List of length 0
call	language	lm(formula = lifeExp ~ year, data = cana...
terms	formula	lifeExp ~ year
model	list [12 x 2] (S3: data.frame)	A data.frame with 12 rows and 2 columns

But the object has everything inside you might need to calculate useful summaries

`summary(canada_lm)`



```
Call:
lm(formula = lifeExp ~ year, data = canada)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.3812 -0.1368 -0.0471  0.2481  0.3157 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -3.583e+02  8.252e+00  -43.42 1.01e-12 ***
year         2.189e-01  4.169e-03   52.50 1.52e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2492 on 10 degrees of freedom
Multiple R-squared:  0.9964,	Adjusted R-squared:  0.996 
F-statistic:  2757 on 1 and 10 DF,  p-value: 1.521e-13
```

A more comprehensive summary,

but not easy to work with.

broom

broom

Turns model output into data frames

```r
# install.packages("tidyverse")
library(broom)
```

broom

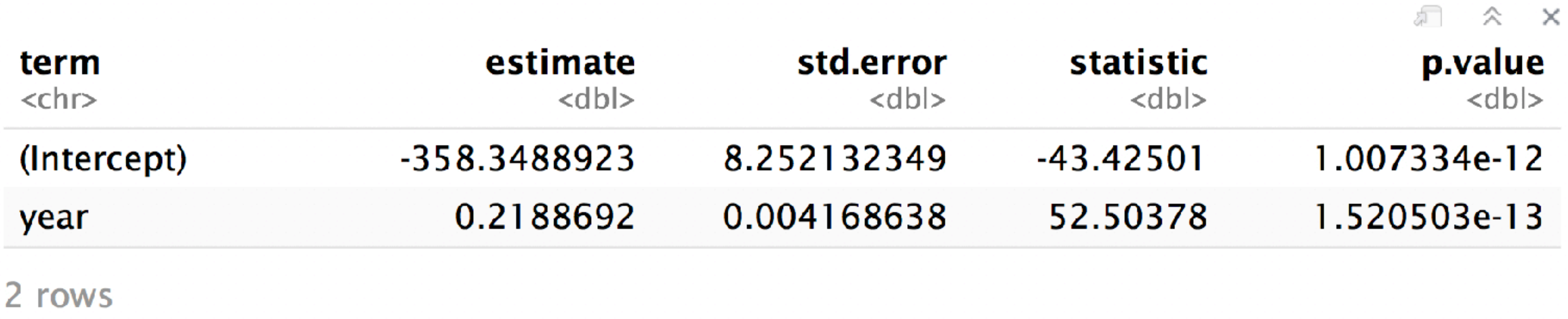
Broom includes three functions which work for most types of models (and can be extended to more):

1. **tidy()** - returns model coefficients, stats
2. **glance()** - returns model diagnostics
3. **augment()** - returns predictions, residuals, and other raw values

tidy()

Returns useful **model output** as a data frame

```
canada_lm %>% tidy()
```



term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	-358.3488923	8.252132349	-43.42501	1.007334e-12
year	0.2188692	0.004168638	52.50378	1.520503e-13

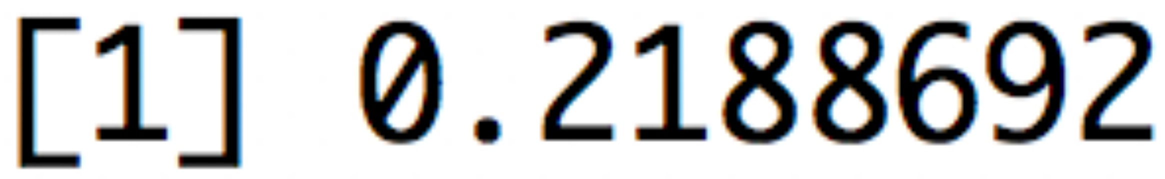
2 rows

Adapted from 'Master the tidyverse' CC by RStudio

Easy to work with

Since output of tidy is a data frame we can use dplyr functions to manipulate it:

```
canada_lm %>% tidy() %>%
  filter(term == "year") %>%
  pull(estimate)
```



```
[1] 0.2188692
```

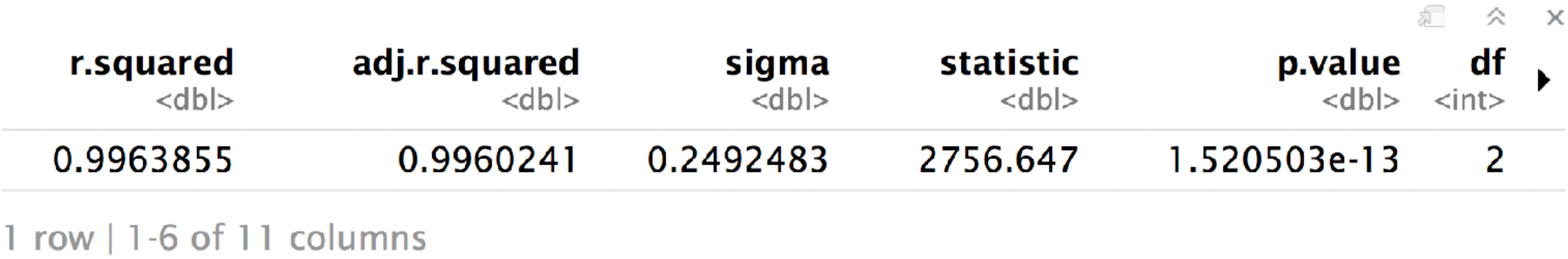
On average, life expectancy is increasing by 0.22 years per year.

Adapted from 'Master the tidyverse' CC by RStudio

glance()

Returns common **model diagnostics** as a data frame

```
canada_lm %>% glance()
```



r.squared	adj.r.squared	sigma	statistic	p.value	df
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
0.9963855	0.9960241	0.2492483	2756.647	1.520503e-13	2

1 row | 1-6 of 11 columns

CC by RStudio

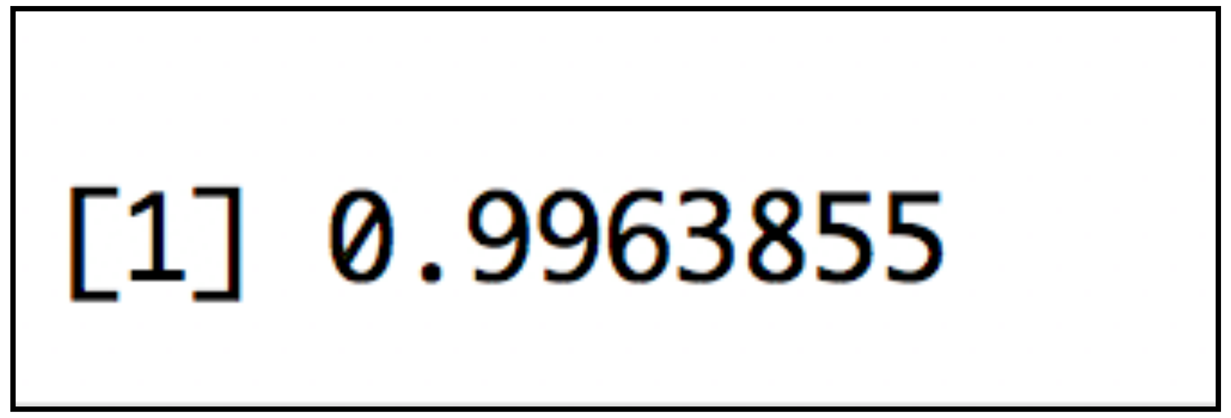
Your Turn 3

Look at the output of `canada_lm %>% glance()`

Use dplyr tools to extract the R-squared for the model.

05:00

```r
canada_lm %>% glance() %>% pull(r.squared)
```

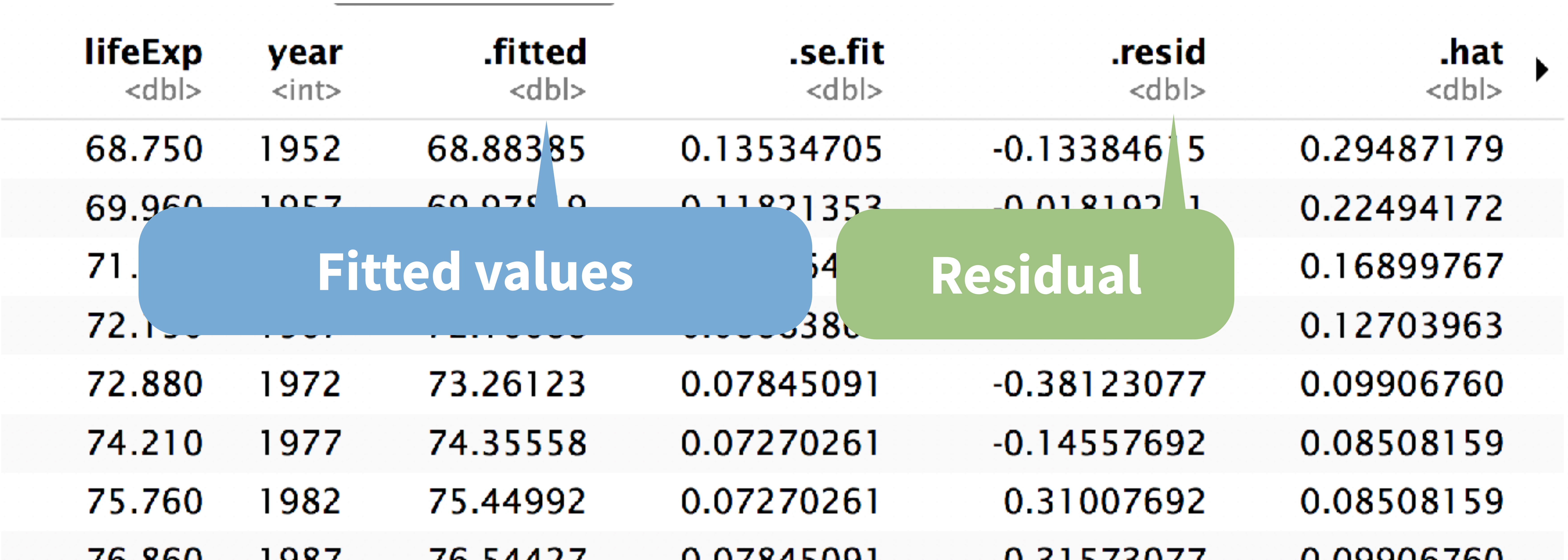


```
[1] 0.9963855
```

augment()

Returns data frame of **model output related to original data points**

```
canada_lm %>% augment()
```



lifeExp <dbl>	year <int>	.fitted <dbl>	.se.fit <dbl>	.resid <dbl>	.hat <dbl>
68.750	1952	68.88385	0.13534705	-0.13384615	0.29487179
69.960	1957	[illegible]	0.11821353	[illegible]	0.22494172
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.16899767
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.12703963
72.880	1972	73.26123	0.07845091	-0.38123077	0.09906760
74.210	1977	74.35558	0.07270261	-0.14557692	0.08508159
75.760	1982	75.44992	0.07270261	0.31007692	0.08508159
76.860	1987	76.54427	0.07845091	0.31573077	0.09906760

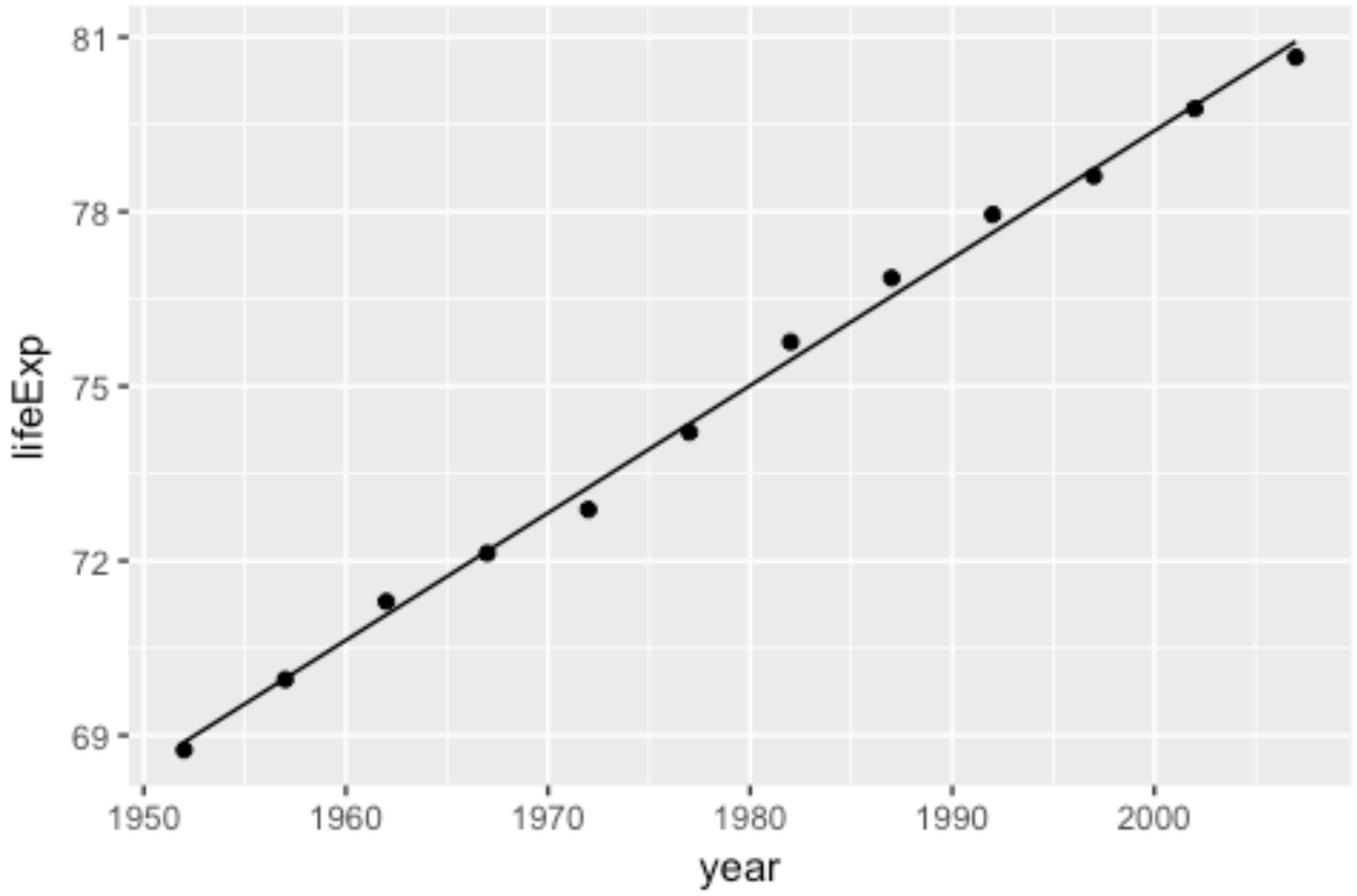
Fitted values

Residual

CC by RStudio

augment() useful for visualizing

```
canada_lm %>% augment() %>%
ggplot() +
  geom_point(aes(x = year, y = lifeExp)) +
  geom_line(aes(x = year, y = .fitted))
```



Adapted from 'Master the tidyverse' CC by RStudio

Recap

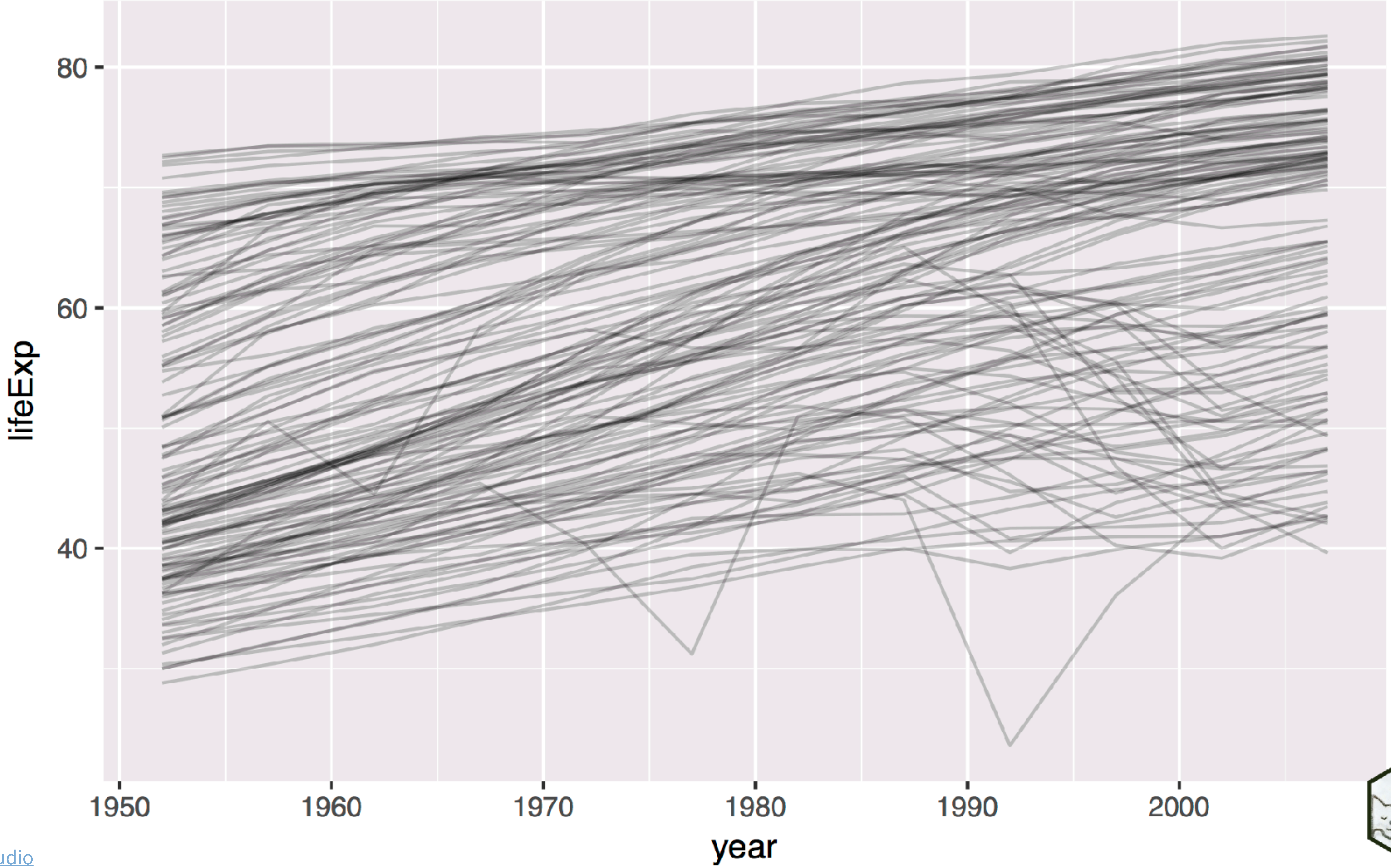
Use **glance()**, **tidy()**, and **augment()** to return model values in a data frame.

Your Turn 4

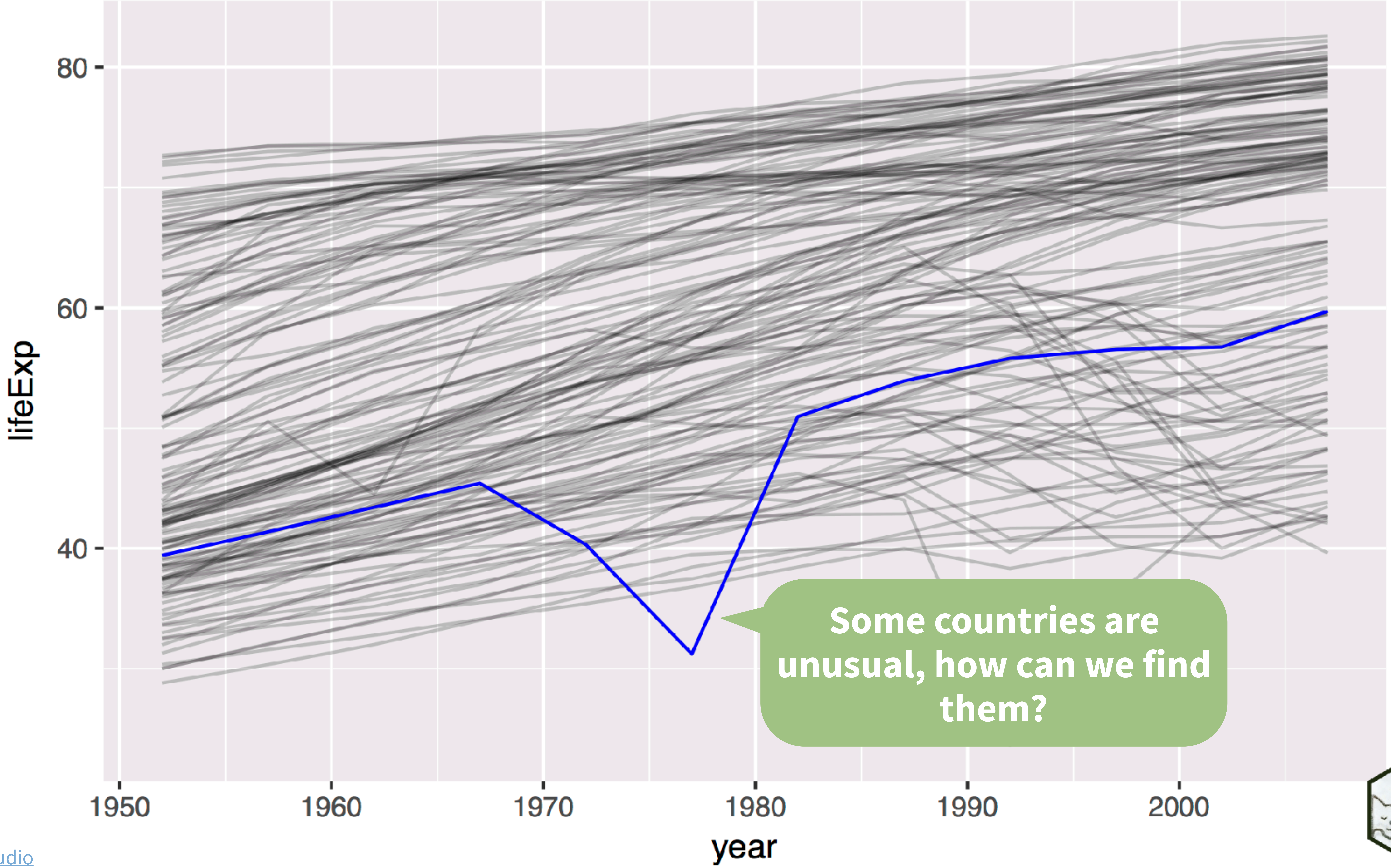
How has life expectancy changed in other countries?

Make a line plot of **lifeExp** vs. **year** grouped by **country**. Set alpha to 0.2, to see the results better.

```
gapminder %>%
  ggplot(mapping = aes(x = year, y = lifeExp, group = country)) +
    geom_line(alpha = 0.2)
```



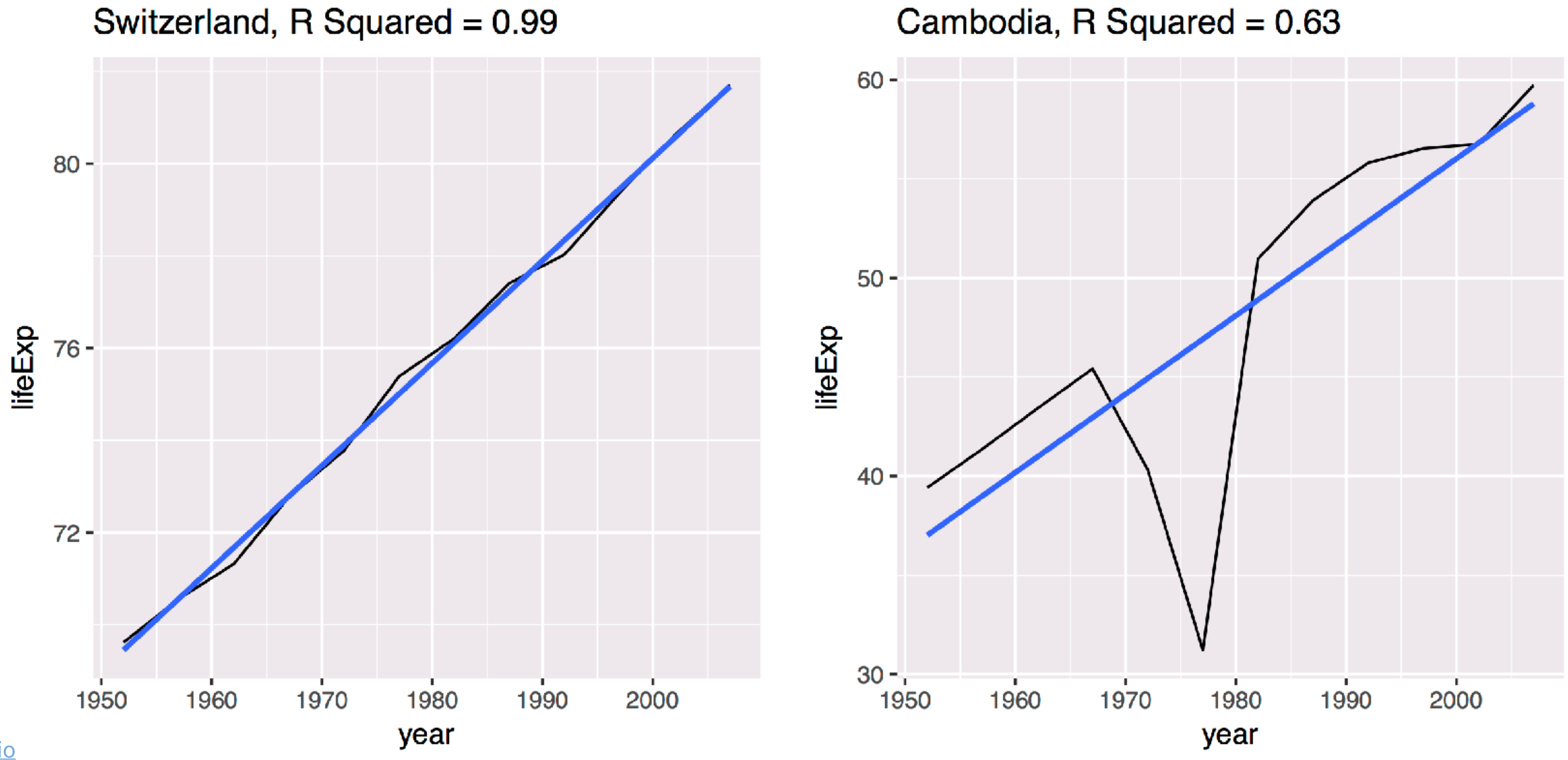
CC by RStudio



Some countries are unusual, how can we find them?

Idea 1

To quantify "linearity," fit a linear model, compare **r-squared**.

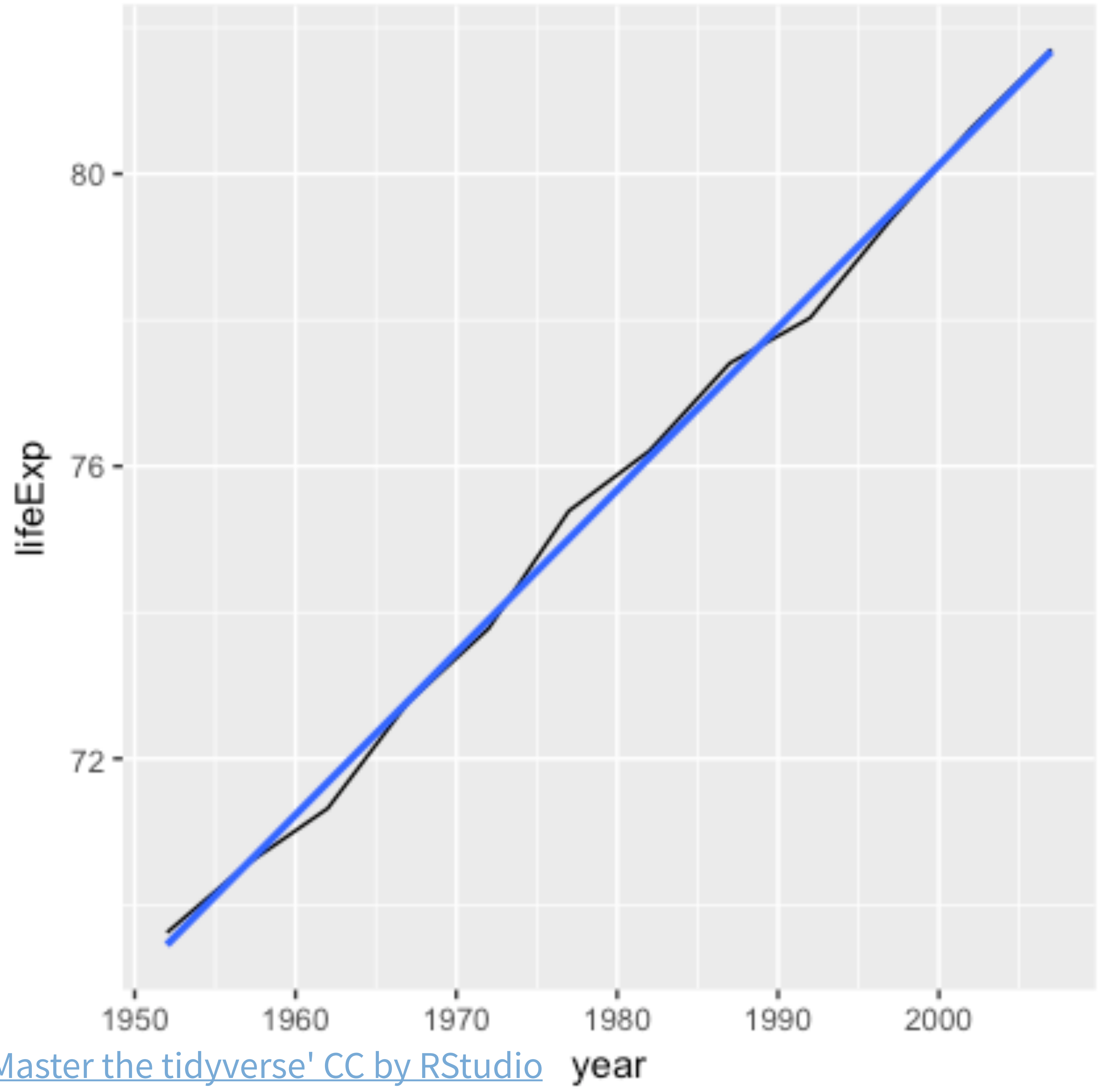


CC by RStudio

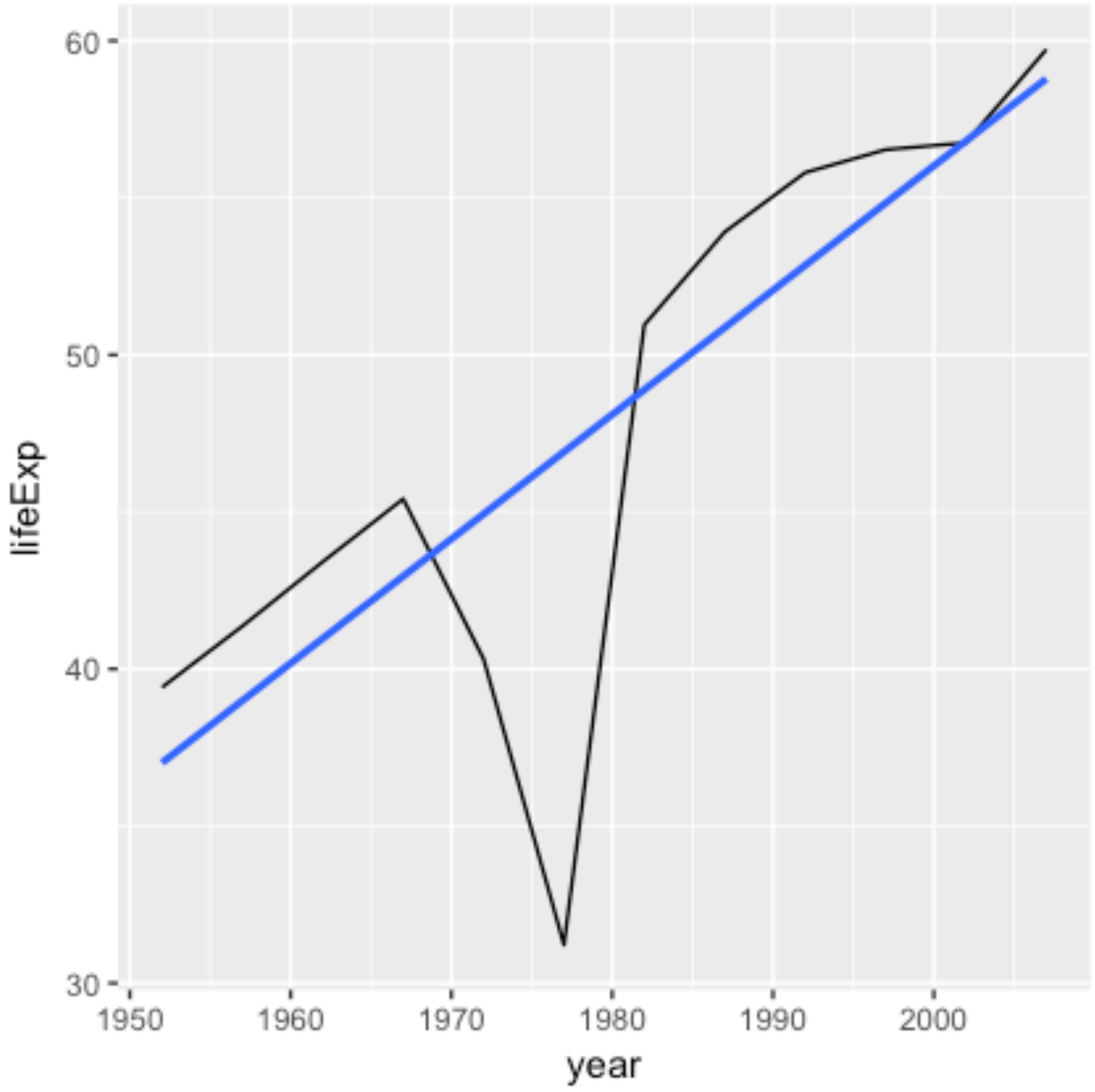
Idea 2

To quantify rate of change fit a linear model, extract **coefficient on year**.

Switzerland = 0.22 years/year



Cambodia = 0.40 years/year



Adapted from 'Master the tidyverse' CC by RStudio

Goal

Fit model, compute r.squared, collect coefficient ***for every country.***

1. **dplyr** grouping toolkit
2. **purrr** toolkit and list columns