Regression demo

```r
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0

## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```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
```

Example 1

We want to predict `lifeExp` using `year` and `gdpPercap`.

```r
fit <- lm(lifeExp ~ year + gdpPercap, data = gapminder)
fit
```

```
##
## Call:
## lm(formula = lifeExp ~ year + gdpPercap, data = gapminder)
##
## Coefficients:
## (Intercept)         year    gdpPercap
##  -4.184e+02    2.390e-01    6.697e-04
```

confidence intervals of coefficients

```r
confint(fit)
```

```
##                     2.5 %        97.5 %
## (Intercept) -4.725914e+02 -3.642571e+02
## year         2.115805e-01  2.663851e-01
## gdpPercap    6.217371e-04  7.177274e-04
```

```r
confint(fit, "year")
```

```
##          2.5 %    97.5 %
## year 0.2115805 0.2663851
```

```r
confint(fit, 2)
```

```
##          2.5 %    97.5 %
## year 0.2115805 0.2663851
```

```r
confint(fit, level = 0.9)
```

```
##                       5 %          95 %
## (Intercept) -4.638752e+02 -3.729734e+02
## year         2.159899e-01  2.619756e-01
## gdpPercap    6.294602e-04  7.100043e-04
```

```r
summary(fit)
```

```
##
## Call:
## lm(formula = lifeExp ~ year + gdpPercap, data = gapminder)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -67.262  -6.954   1.219   7.759  19.553
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4.184e+02  2.762e+01  -15.15   <2e-16 ***
## year         2.390e-01  1.397e-02   17.11   <2e-16 ***
## gdpPercap    6.697e-04  2.447e-05   27.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.694 on 1701 degrees of freedom
## Multiple R-squared:  0.4375, Adjusted R-squared:  0.4368
## F-statistic: 661.4 on 2 and 1701 DF,  p-value: < 2.2e-16
```

```r
library(broom)
glance(fit)
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic   p.value    df  logLik    AIC
## *     <dbl>         <dbl> <dbl>     <dbl>     <dbl> <int>   <dbl>  <dbl>
## 1     0.437         0.437  9.69      661. 3.13e-213     3 -6287. 12582.
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

Comparing models via adjusted R^2

```
fit0 <- lm(lifeExp ~ year, data = gapminder)
fit2 <- lm(lifeExp ~ year + pop, data = gapminder)
fit3 <- lm(lifeExp ~ year + gdpPercap + pop, data = gapminder)
```

```
bind_rows(
    glance(fit0),
    glance(fit),
    glance(fit2),
    glance(fit3)
)
```

```
## # A tibble: 4 x 11
##   r.squared adj.r.squared sigma statistic   p.value    df  logLik    AIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <int>   <dbl>  <dbl>
## 1     0.190         0.189 11.6       399. 7.55e- 80     2 -6598. 13202.
## 2     0.437         0.437  9.69      661. 3.13e-213     3 -6287. 12582.
## 3     0.191         0.190 11.6       200. 7.73e- 79     3 -6597. 13202.
## 4     0.440         0.439  9.67      446. 1.52e-213     4 -6283. 12576.
## # ... with 3 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>
```

Assessing the Accuracy of the estimated regression model

```
library(modelr)
```

```
##
## Attaching package: 'modelr'

## The following object is masked from 'package:broom':
##
##     bootstrap
```

```
rp <- resample_partition(gapminder, c(train = 0.7, test = 0.3))
rp
```

```
## $train
## <resample [1,192 x 6]> 1, 3, 4, 5, 7, 8, 9, 12, 13, 14, ...
##
## $test
## <resample [512 x 6]> 2, 6, 10, 11, 16, 22, 23, 26, 39, 40, ...
```

```r
training_set <- as.tibble(rp$train)
testing_set <- as.tibble(rp$test)
```

```r
fit0 <- lm(lifeExp ~ year, data = training_set)
fit1 <- lm(lifeExp ~ year + gdpPercap, data = training_set)
fit2 <- lm(lifeExp ~ year + pop, data = training_set)
fit3 <- lm(lifeExp ~ year + gdpPercap + pop, data = training_set)
```

```r
c(mse(fit0, testing_set),
  mse(fit1, testing_set),
  mse(fit2, testing_set),
  mse(fit3, testing_set))
```

```
## [1] 137.3380 104.3193 137.0711 103.7084
```

Interpreting regression coefficients

```r
supermodel <- read_tsv("supermodel.dat")
```

```
## Parsed with column specification:
## cols(
##   salary = col_double(),
##   age = col_double(),
##   years = col_double(),
##   beauty = col_double()
## )
```

```r
supermodel
```

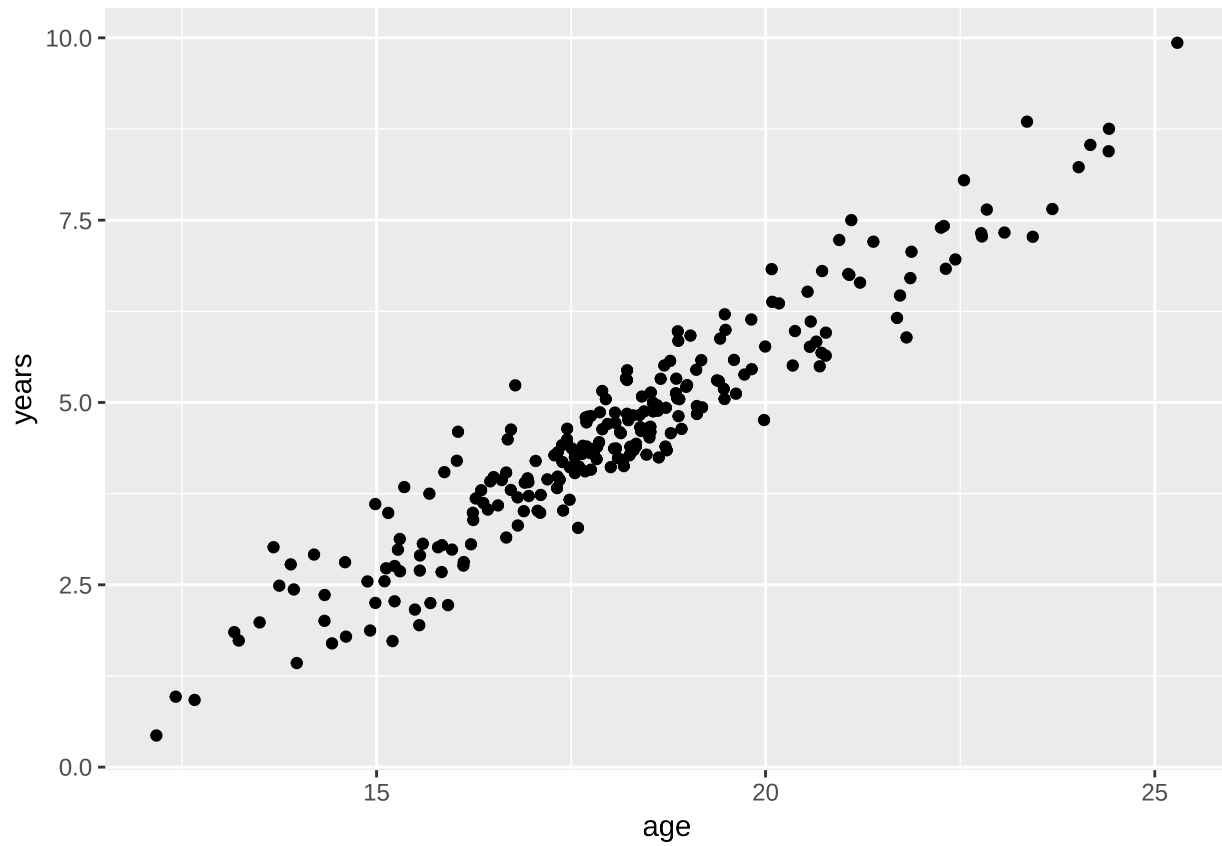
```
## # A tibble: 231 x 4
##     salary   age years beauty
##      <dbl> <dbl> <dbl>  <dbl>
##  1  0.370   16.7  3.15   78.3
##  2 53.7     20.3  5.51   68.6
##  3  1.46    18.2  5.33   75.0
##  4  0.0243  15.4  3.84   65.1
##  5 95.3     24.2  8.53   71.8
##  6 14.6     18.3  4.39   78.1
##  7  8.67    17.7  4.40   72.1
##  8  2.65    17.5  4.11   75.3
##  9  7.55    17.1  3.52   72.0
## 10  1.20    20.1  6.83   71.9
## # ... with 221 more rows
```

```r
(fit <- lm(salary ~ age + years + beauty, data = supermodel))
```

```
##
## Call:
```

```
## lm(formula = salary ~ age + years + beauty, data = supermodel)
##
## Coefficients:
## (Intercept)          age        years       beauty
##    -60.8897       6.2344      -5.5612      -0.1964
```

```r
ggplot(supermodel) + geom_point(aes(x = age, y = years))
```



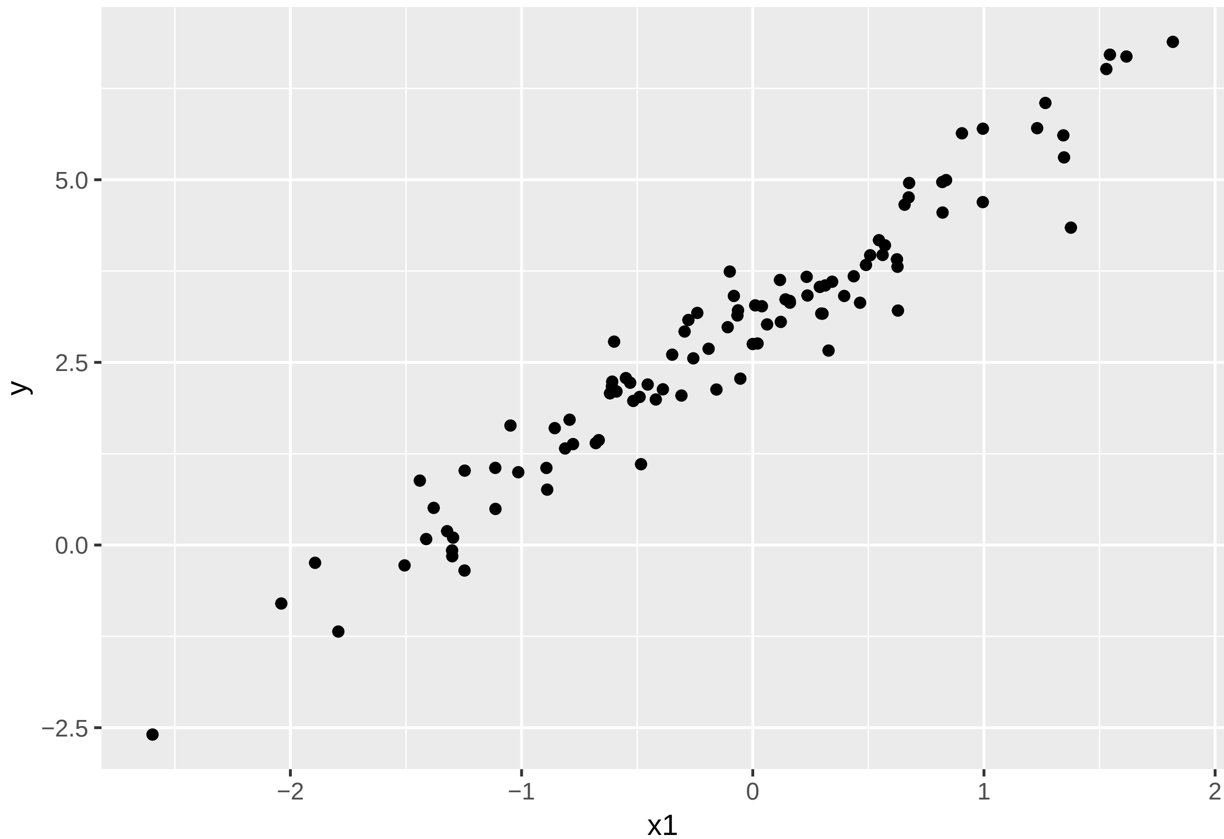
Correlations among preditors

```r
x1 <- rnorm(100)
x2 <- 2 * x1 + rnorm(100, sd = 0.1)
y <- 3 + 2 * x1 + rnorm(100, sd = 0.5)
(example <- tibble(y = y, x1 = x1, x2 = x2))
```

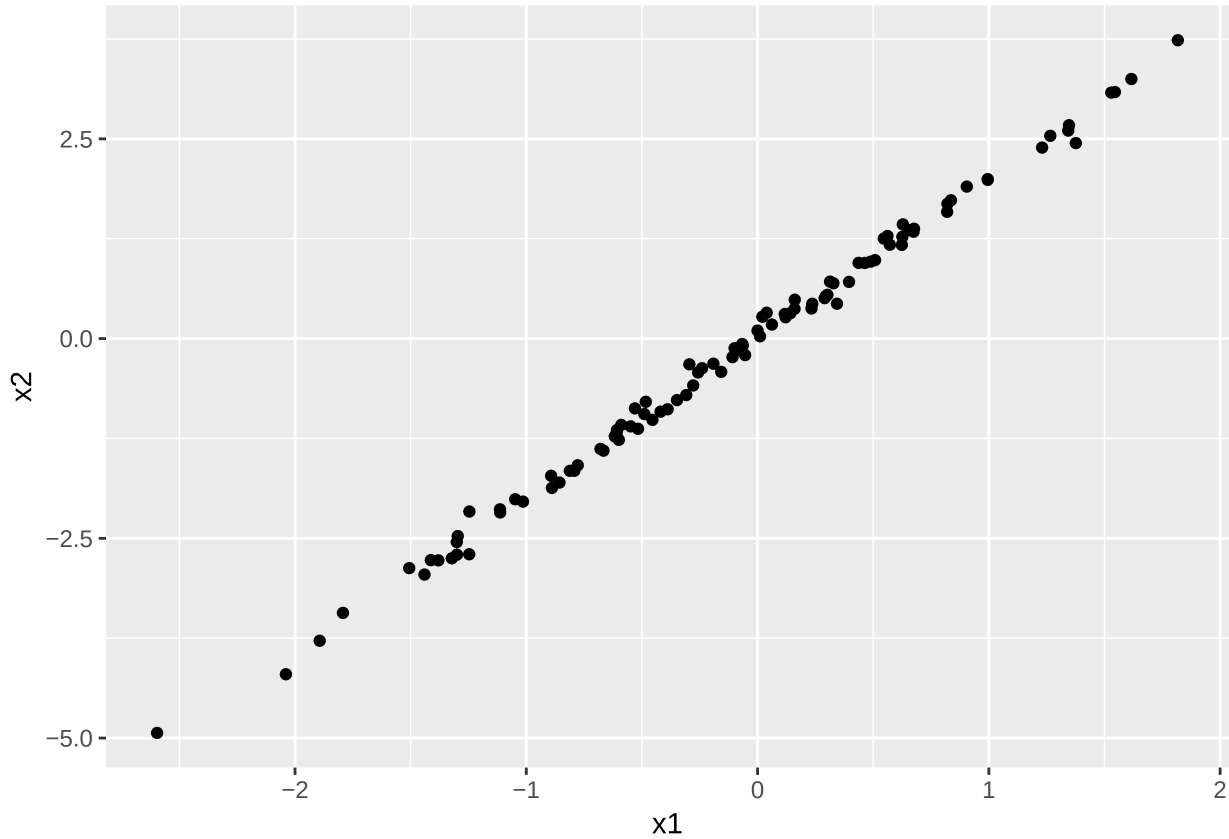
```
## # A tibble: 100 x 3
##        y      x1     x2
##    <dbl>   <dbl>  <dbl>
##  1 5.64   0.905   1.90
##  2 3.02   0.0620  0.176
##  3 1.32  -0.812  -1.66
##  4 3.91   0.624   1.17
##  5 2.13  -0.389  -0.885
```

```
##   6 0.758 -0.889  -1.87
##   7 1.64  -1.05   -2.01
##   8 2.05  -0.308  -0.707
##   9 2.22  -0.530  -0.874
##  10 1.99  -0.419  -0.914
## # ... with 90 more rows
```

```
ggplot(example) + geom_point(aes(x1, y))
```




```
ggplot(example) + geom_point(aes(x1, x2))
```




```r
(fit <- lm(y ~ x1 + x2, data = example))
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = example)
##
## Coefficients:
## (Intercept)           x1           x2
##      3.0164       0.5190       0.7541
```

Non linear transformation of the predictors

```r
rp <- resample_partition(gapminder, c(train = 0.7, test = 0.3))
rp
```

```
## $train
## <resample [1,192 x 6]> 1, 2, 3, 6, 7, 8, 10, 11, 15, 18, ...
##
## $test
## <resample [512 x 6]> 4, 5, 9, 12, 13, 14, 16, 17, 22, 24, ...
```

```r
training_set <- as.tibble(rp$train)
testing_set <- as.tibble(rp$test)
fit_linear <- lm(lifeExp ~ gdpPercap, data = training_set)
fit_quad <- lm(lifeExp ~ gdpPercap + I(gdpPercap^2), data = training_set)
```

Which one is better?

```r
c(mse(fit_linear, testing_set), mse(fit_quad, testing_set))
```

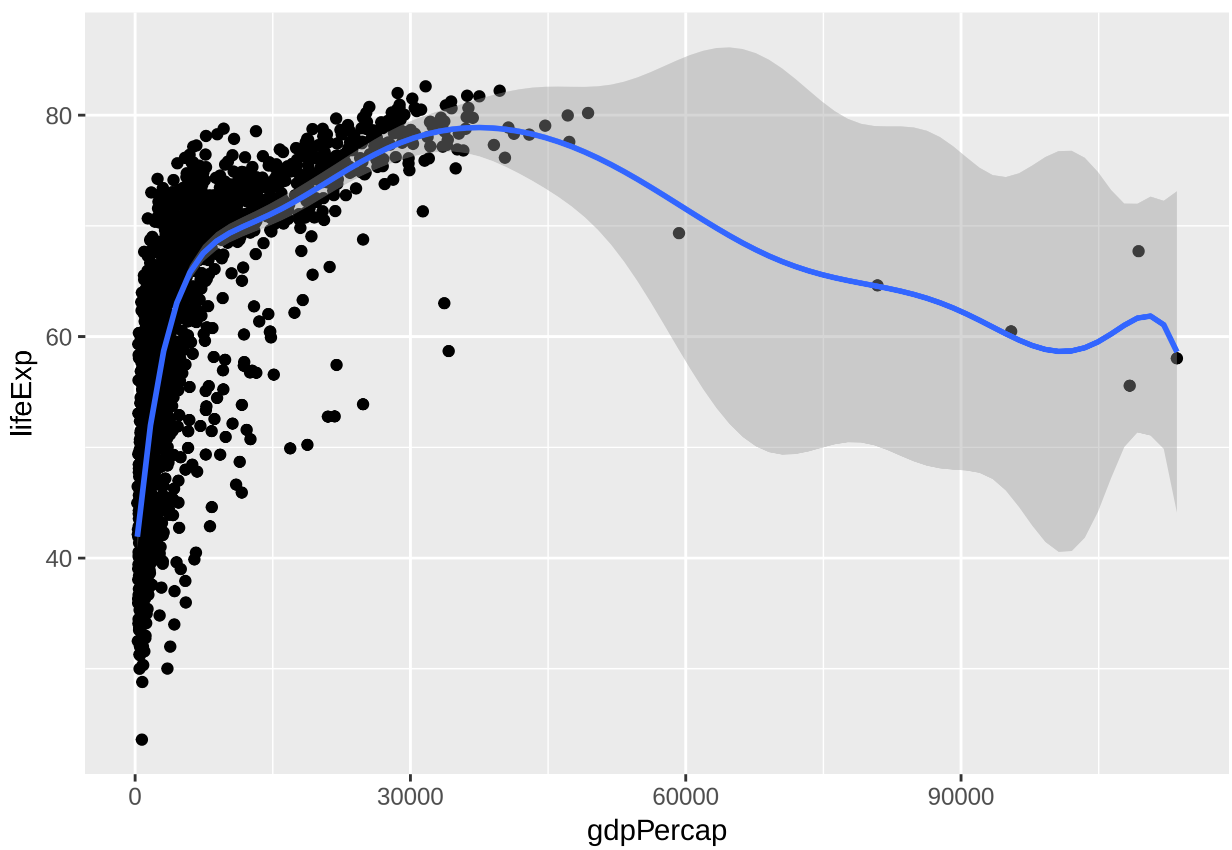
```
## [1] 100.33030  80.03267
```

How about adding a cubic term? Adding `I(gdpPercap^3)` term? No. Use `poly`!

```r
fit_cubic <- lm(lifeExp ~ poly(gdpPercap, 3), data = training_set)
mse(fit_cubic, testing_set)  # even better
```

```
## [1] 67.38276
```

```r
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) + geom_point() +
    geom_smooth(method = "lm", formula = y ~ poly(x, 10))
```



Let's be crazy

```r
fit_poly <- lm(lifeExp ~ poly(gdpPercap, 8), data = training_set)
mse(fit_poly, testing_set)  # becoming worse
```

```
## [1] 54.29174
```

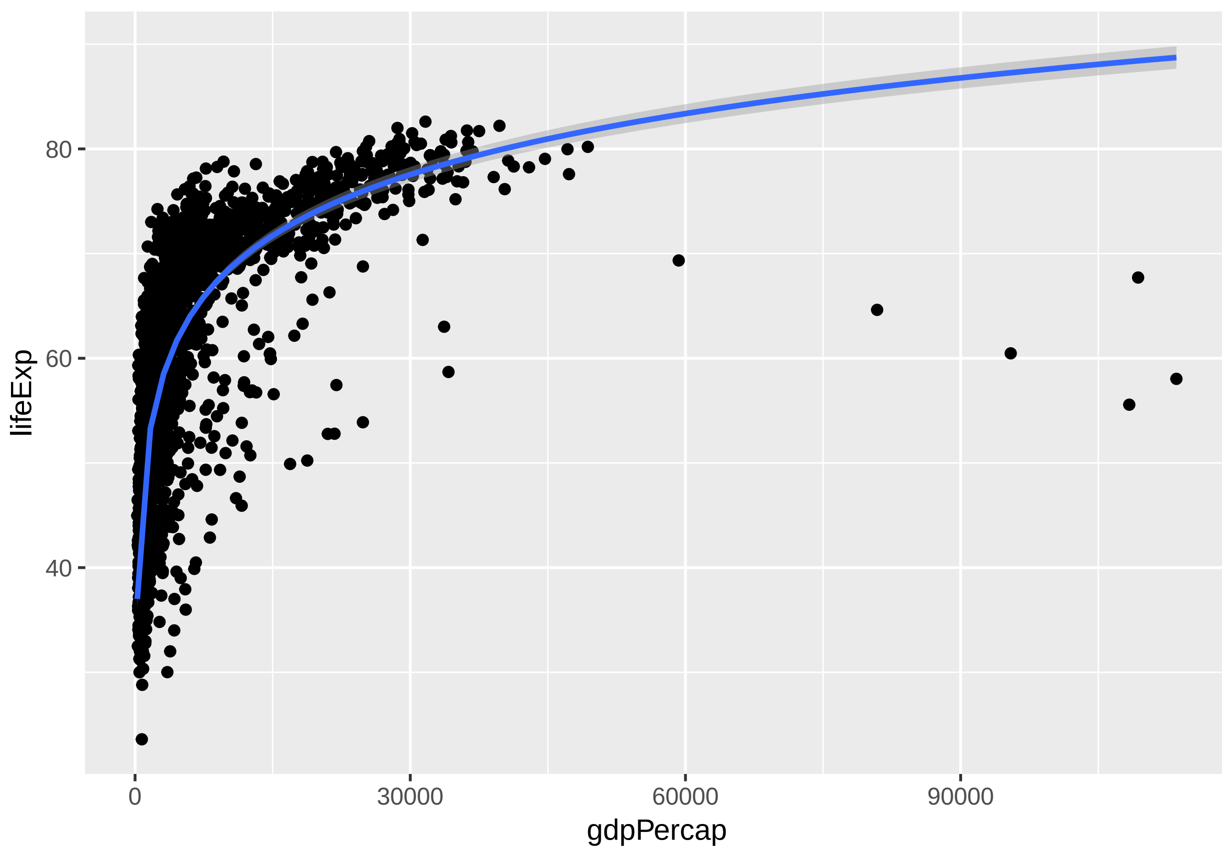
Of course, we are not limited to polynomials

```r
fit_log <- lm(lifeExp ~ log(gdpPercap), data = training_set)
mse(fit_log, testing_set)  # wow, even better
```

```
## [1] 56.3479
```

In fact, look at this

```r
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) + geom_point() +
    geom_smooth(method = "lm", formula = y ~ log(x))
```



Qualitative Predictor

```r
gapminder %>% distinct(country)
```

```
## # A tibble: 142 x 1
##    country
##    <fct>
##  1 Afghanistan
##  2 Albania
##  3 Algeria
##  4 Angola
##  5 Argentina
##  6 Australia
##  7 Austria
##  8 Bahrain
##  9 Bangladesh
## 10 Belgium
## # ... with 132 more rows
```

Let's consider a smaller

```r
little_gapminder <- gapminder %>% filter(str_detect(country, "^H")) %>%
    mutate(country = fct_drop(country))
little_gapminder %>% distinct(country)
```

```
## # A tibble: 4 x 1
##   country
##   <fct>
## 1 Haiti
## 2 Honduras
## 3 Hong Kong, China
## 4 Hungary
```

Now, we want to use `country` to predict `lifeExp`.

```r
(fit <- lm(lifeExp ~ country, data = little_gapminder))
```

```
##
## Call:
## lm(formula = lifeExp ~ country, data = little_gapminder)
##
## Coefficients:
##             (Intercept)          countryHonduras  countryHong Kong, China
##                  50.165                    7.756                   23.328
##          countryHungary
##                  19.228
```

Under the hood.

```r
contrasts(little_gapminder$country)
```

```
##                  Honduras Hong Kong, China Hungary
## Haiti                   0                0       0
## Honduras                1                0       0
## Hong Kong, China        0                1       0
## Hungary                 0                0       1
```

A common mistake

```r
(example2 <- tibble(x = sample(1:3, 10, replace = TRUE), y = rnorm(10)))
```

```
## # A tibble: 10 x 2
##        x       y
##    <int>   <dbl>
##  1     2  1.42
##  2     1  0.0429
##  3     1 -0.456
##  4     3 -0.518
##  5     2 -0.840
##  6     3  0.860
##  7     3 -1.31
##  8     3 -0.239
##  9     2  0.420
## 10     2  0.991
```

```r
lm(y ~ x, data = example2)  # wrong
```

```
##
## Call:
## lm(formula = y ~ x, data = example2)
##
## Coefficients:
## (Intercept)            x
##      0.3785      -0.1551
```

```r
example2_corrected <- example2 %>% mutate(x = recode_factor(x, `1` = "blue", `2` = "green", `3` = "red"
example2_corrected
```

```
## # A tibble: 10 x 2
##    x           y
##    <fct>   <dbl>
##  1 green  1.42
##  2 blue   0.0429
##  3 blue  -0.456
##  4 red   -0.518
##  5 green -0.840
##  6 red    0.860
##  7 red   -1.31
##  8 red   -0.239
##  9 green  0.420
## 10 green  0.991
```

```r
lm(y ~ x, data = example2_corrected)  # correct
```

```
##
## Call:
## lm(formula = y ~ x, data = example2_corrected)
```

```
##
## Coefficients:
## (Intercept)       xgreen         xred
##    -0.20632      0.70426     -0.09535
```

```
contrasts(example2_corrected$x)
```

```
##       green red
## blue      0   0
## green     1   0
## red       0   1
```

Prediction

```
fit <- lm(lifeExp ~ year, data = gapminder)
new_data <- tibble(year = 100)

# the classic way
predict(fit, new_data)
```

```
##         1
## -553.0618
```

```
# in tidyverse style
library(modelr)
new_data %>% add_predictions(fit)
```

```
## # A tibble: 1 x 2
##    year  pred
##   <dbl> <dbl>
## 1   100 -553.
```