Week 4 Summary

Leo Soccio

Table of contents

Tuesday, Jan 17 1
Statistical Learning 2
Thursday, Jan 19 6

Tuesday, Jan 17

> **! TIL**
>
> Include a *very brief* summary of what you learnt in this class here.
> Today, I learnt the following concepts in class:
>
> 1. Intro to Statistical Learning
> 2. How the Linear Regression Model is Created

Provide more concrete details here. You can also use footnotes[1] if you like

agenda: basics of statistical learning + regression

```r
#required packages
library(tidyverse)
```

```
-- Attaching packages --------------------------------------- tidyverse 1.3.2 --
v ggplot2 3.4.0     v purrr   1.0.1
v tibble  3.1.8     v dplyr   1.1.0
```

[1] You can include some footnotes here

```
v tidyr   1.3.0     v stringr 1.5.0
v readr   2.1.3     v forcats 1.0.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

```r
library(ISLR2)
library(cowplot)
library(kableExtra)
```

```
Attaching package: 'kableExtra'

The following object is masked from 'package:dplyr':

    group_rows
```

```r
library(htmlwidgets)
```

Statistical Learning

Suppose we have a data set: $X = [X_1, X_2 ... X_p]$ Each X_n is a covariate (also predictor variable or independent variable). Then we have y, the response/outcome/dependent variable.

Statistical learning is to find a function f such that y=f(X) such that $y_i = f(X_i) = f(X_{i1} ... X_{ip})$

We should have a way to map covariates to the response. There are different flavors of statistical learning:

- Supervised Learning (Includes **regression** [for quantitative y] and classification [for categorical y])
- Unsupervised Learning (no y, we need to figure out what it could be)
- Semi-supervised learning (We have far more total observations than observations including a y-value)
- Reinforcement learning (the algorithm is "punished" for doing something "wrong")

We will focus on regression today. We will start with an example from the U.S. Census regarding teen birth rate and poverty in each state.

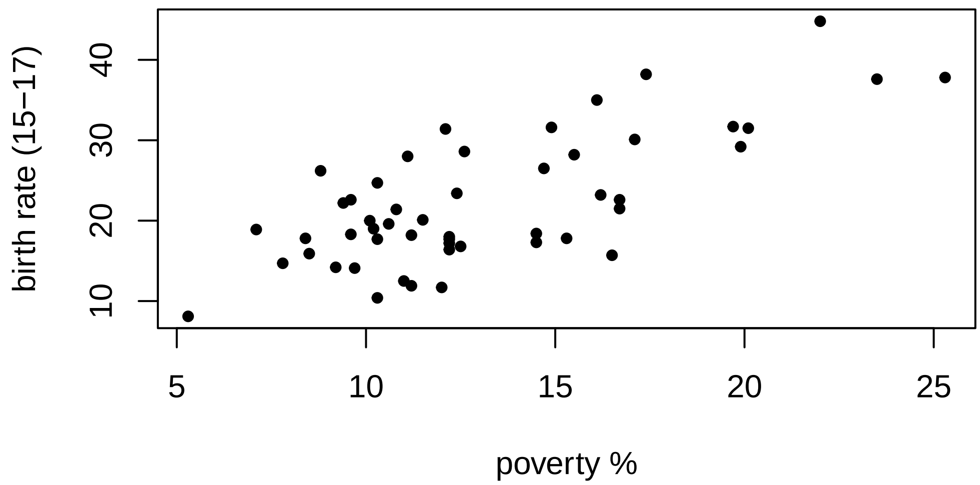
```r
# load in the data
df <- read_tsv("https://online.stat.psu.edu/stat462/sites/onlinecourses.science.psu.edu.st
```

```
Rows: 51 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: "\t"
chr (1): Location
dbl (5): PovPct, Brth15to17, Brth18to19, ViolCrime, TeenBrth

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(df, 10)
```

```
# A tibble: 10 x 6
   Location             PovPct Brth15to17 Brth18to19 ViolCrime TeenBrth
   <chr>                 <dbl>      <dbl>      <dbl>     <dbl>    <dbl>
 1 Alabama                20.1       31.5       88.7      11.2     54.5
 2 Alaska                  7.1       18.9       73.7       9.1     39.5
 3 Arizona                16.1       35        102.       10.4     61.2
 4 Arkansas               14.9       31.6      102.       10.4     59.9
 5 California             16.7       22.6       69.1      11.2     41.1
 6 Colorado                8.8       26.2       79.1       5.8     47
 7 Connecticut             9.7       14.1       45.1       4.6     25.8
 8 Delaware               10.3       24.7       77.8       3.5     46.3
 9 District_of_Columbia   22         44.8      102.       65       69.1
10 Florida                16.2       23.2       78.4       7.3     44.5
```

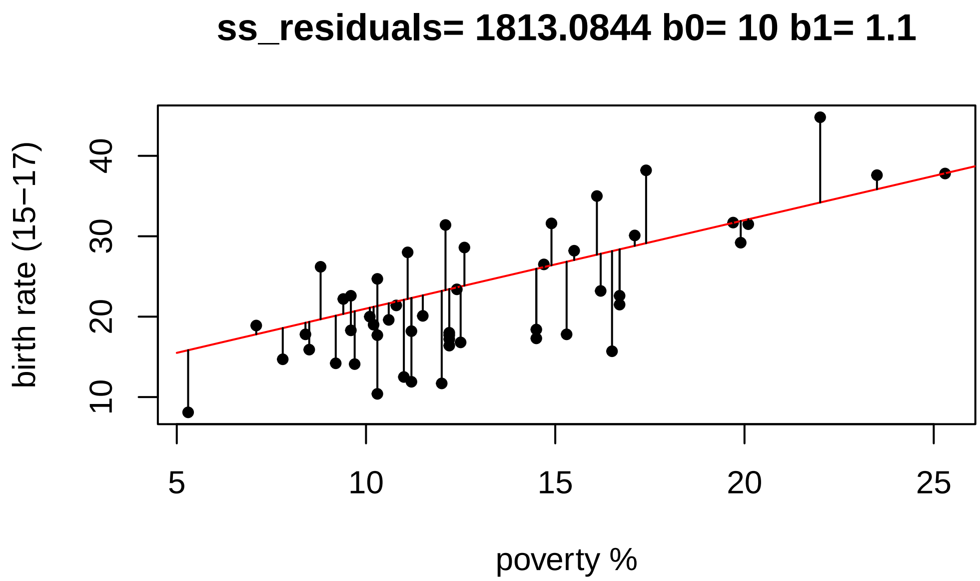
```r
# define our covariate and our response, then visualize the relationship
x <- df$PovPct
y <- df$Brth15to17
plot(x,y,pch=20, xlab="poverty %", ylab="birth rate (15-17)")
```



To create a linear regression curve, we want to fit a regression line $y = \beta_0 + \beta_1 x$.

Create a line through the points:

```
plt <- function(){
  plot(x,y,pch=20,xlab="poverty %", ylab="birth rate (15-17)")
}

b0 <- 10
b1 <- 1.1

yhat<-b0+b1*x

plt()
curve(b0+b1*x, 5, 30, add=T, col="red")
segments(x,y,x,yhat)

resids <- abs(y-yhat)^2
ss_resids <- sum(resids)
title(main=paste("ss_residuals=",ss_resids,"b0=",b0,"b1=",b1))
```



Least squares regression is calculated by dropping a vertical line (residual=$y - \hat{y}$) from each data point to the fit line. The residual is then squared and those squared residuals are summed to get a sum of squares. We want to find the line with the lowest sum of squares.

The lm() function creates this model for us in R.

```
model <- lm(y~x)

sum(residuals(model)^2)
```

```
[1] 1509.635
```

```
summary(model)
```

```
Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-11.2275  -3.6554  -0.0407   2.4972  10.5152
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.2673     2.5297   1.687    0.098 .
x             1.3733     0.1835   7.483 1.19e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.551 on 49 degrees of freedom
Multiple R-squared:  0.5333,	Adjusted R-squared:  0.5238
F-statistic:    56 on 1 and 49 DF,  p-value: 1.188e-09
```

Thursday, Jan 19

! TIL

Include a *very brief* summary of what you learnt in this class here.
Today, I learnt the following concepts in class:

1. Details of Linear Regression (hypotheses, p-values, beta variables, etc.)
2. R-squared
3. Predicting using the linear regression model

Provide more concrete details here:

When creating a model, we want y as a function of x. In R this looks like:

```r
formula(y~x)
```

```
y ~ x
```

```r
typeof(formula(y~x))
```

```
[1] "language"
```

A linear regression model in R is called using the Linear Model function lm().

```r
model <- lm(y~x)
model
```

```
Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
      4.267        1.373
```

```r
x2<-x^2
model2 <- lm(y~x+x2)
model2
```

```
Call:
lm(formula = y ~ x + x2)

Coefficients:
(Intercept)            x           x2
   10.60211      0.43733      0.03128
```

```r
summary(model)
```

```
Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-11.2275  -3.6554  -0.0407   2.4972  10.5152

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.2673     2.5297   1.687    0.098 .
x             1.3733     0.1835   7.483 1.19e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.551 on 49 degrees of freedom
Multiple R-squared:  0.5333,    Adjusted R-squared:  0.5238
F-statistic:    56 on 1 and 49 DF,  p-value: 1.188e-09
```

What do the hypotheses for regression look like?

- The null hypothesis is that there is no linear relationship between y and x. This means that $\beta_1 = 0$
- The alternate hypothesis is that there is a linear relationship, so $\beta_1 \neq 0$

To summarize, $H_0 : \beta_1 = 0, H_A : \beta_1 \neq 0$

When we see a small p-value, then we reject the null hypothesis in favor of the alternate. This means that there is a significant linear relationship between y and x. That is to say, there is significant evidence of a correlation between x and y.

The p-value at the bottom of the summary is based on the F statistic, which tests the overall model instead of a specific covariate.

** R-Squared**

```
library(broom)

summary(model) %>%
  broom::tidy()
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic       p.value
  <chr>          <dbl>     <dbl>     <dbl>         <dbl>
1 (Intercept)     4.27     2.53       1.69 0.0980
2 x               1.37     0.184      7.48 0.00000000119
```

Some terminology: x is our covariate, y is our response, $\hat{y}$ are the fitted values, and $y - \hat{y}$ are the residuals.

```
head(x)
```

```
[1] 20.1  7.1 16.1 14.9 16.7  8.8
```

```
head(y)
```

```
[1] 31.5 18.9 35.0 31.6 22.6 26.2
```

```
yhat <- fitted(model)
head(yhat)
```

```
       1        2        3        4        5        6 
31.87154 14.01805 26.37815 24.73014 27.20216 16.35273 
```

```
res <- residuals(model)
head(res)
```

```
         1          2          3          4          5          6 
-0.3715352  4.8819549  8.6218464  6.8698609 -4.6021608  9.8472677 
```

Sum of squares for residuals: $SS_{Res} = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

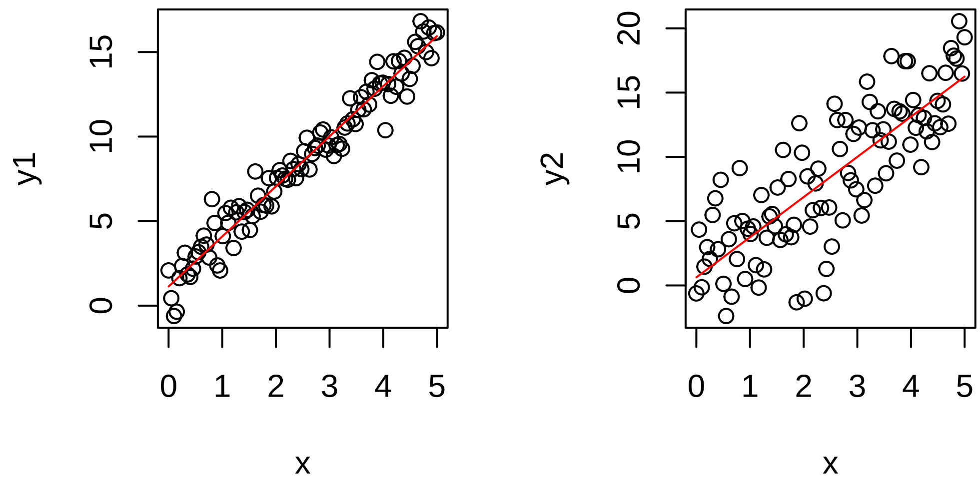
Sum of squares for regression: $SS_{Reg} = \sum_{i=1}^{n} (\hat{y}_i = \bar{y})^2$

Sum of squares total $SS_{Tot} = \sum_{i=1}^{n} (y_i - \bar{y})^2$

R^2 is another important value and is given by $R^2 = \frac{SS_{Reg}}{SS_{Tot}}$

Examples:

```
x <- seq(0,5,length=100)

b0 <-1
b1 <-3

y1 <- b0+b1*x+rnorm(100)
y2 <- b0+b1*x+rnorm(100)*3

par(mfrow=c(1,2)) # lets you create side by side plots. This one is 1 row, 2 cols.

model1 <- lm(y1~x)
model2<-lm(y2~x)

plot(x,y1)
curve(coef(model1)[1]+coef(model1)[2]*x, add=T, col="red")

plot(x, y2)
curve(coef(model2)[1]+coef(model2)[2]*x,add=T,col="red")
```




```r
summary(model1)
```

```
Call:
lm(formula = y1 ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-2.72224 -0.58368  0.02235  0.60560  2.77341

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.13111    0.18558   6.095 2.16e-08 ***
x            2.96212    0.06412  46.193  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9349 on 98 degrees of freedom
Multiple R-squared:  0.9561,    Adjusted R-squared:  0.9556
F-statistic:  2134 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
summary(model2)
```

```
Call:
lm(formula = y2 ~ x)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.6497 -2.1770  0.3655  2.2028  6.1714 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   0.6249     0.6468   0.966    0.336    
x             3.1246     0.2235  13.982   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.258 on 98 degrees of freedom
Multiple R-squared:  0.6661,    Adjusted R-squared:  0.6627 
F-statistic: 195.5 on 1 and 98 DF,  p-value: < 2.2e-16
```

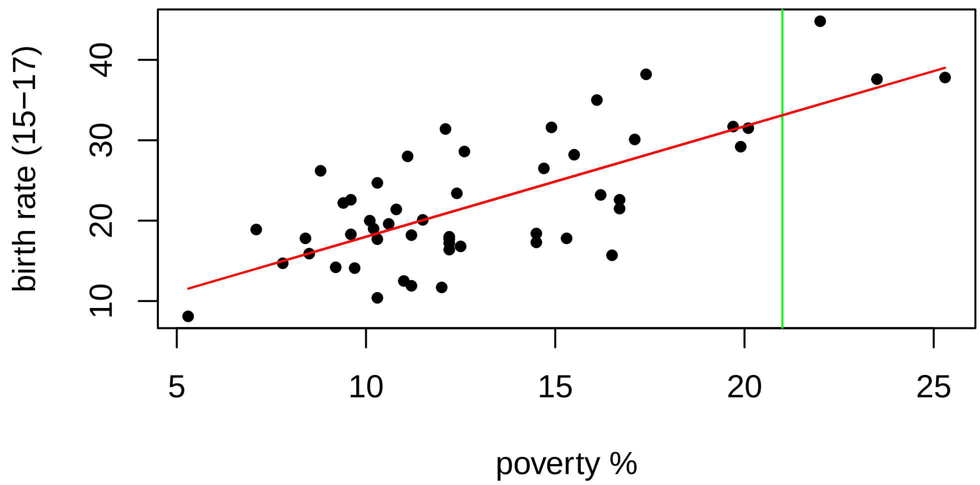
- R-squared and the p-value are independent of each other; just because it is a significant model doesn't mean the model fits closely.

Prediction

Return to the poverty dataset:

Suppose we have a new state formed whose PovPct value is 21:

```r
x <- df$PovPct
y <- df$Brth15to17
plt()
abline(v=21,col="green")
lines(x,fitted(lm(y~x)), col="red")
```

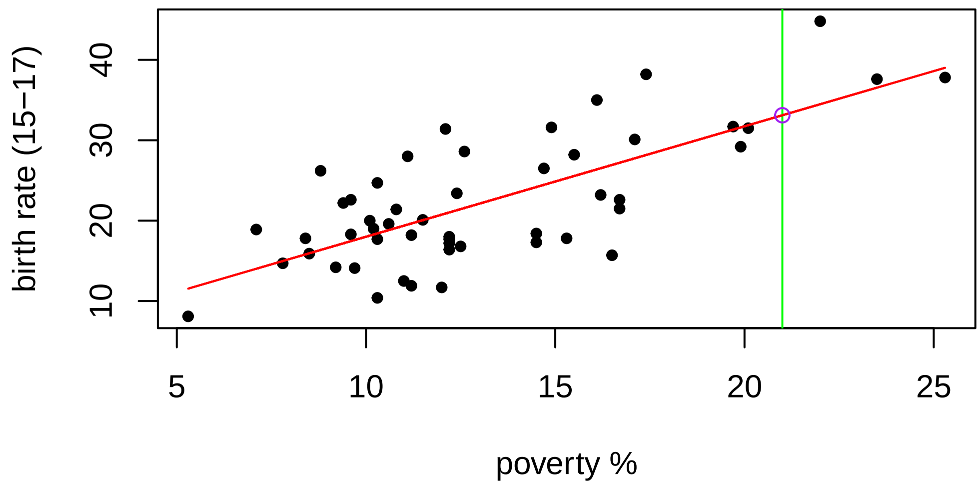


We can look at the regression line to predict the teen birth rate in this state by finding the point on the line where x=21. In R, we can use predict() to do this:

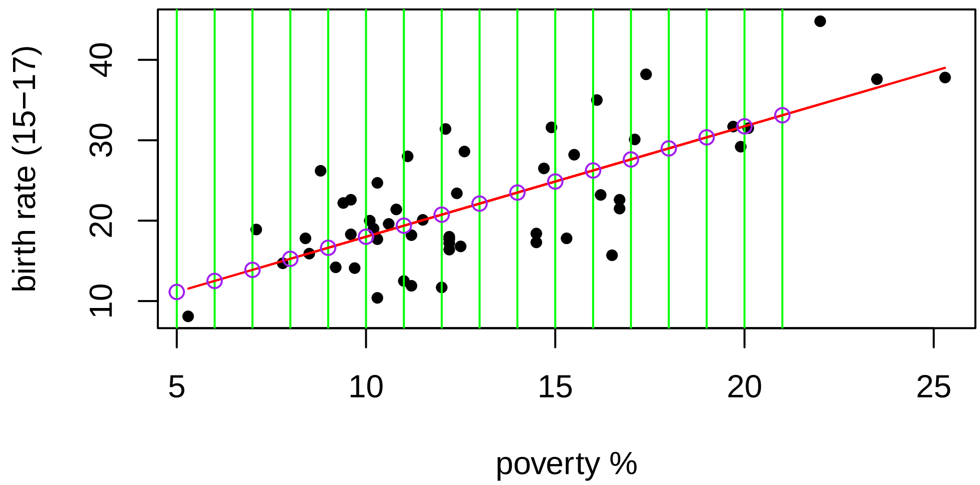
```
model<-lm(y~x)
new_x<-data.frame(x=c(21))
new_y<-predict(model,new_x)
new_y
```

```
       1
33.10755
```

```
plt()
abline(v=21,col="green")
lines(x,fitted(lm(y~x)),col="red")
points(new_x,new_y, col="purple")
```




```
new_x<-data.frame(x=c(1:21))
new_y<-predict(model,new_x)
plt()
for(a in new_x){abline(v=a, col="green")}
lines(x,fitted(lm(y~x)),col="red")
points(new_x%>%unlist(),new_y%>%unlist(),col="purple")
```


birth rate (15−17)
10
20
30
40
5
10
15
20
25
poverty %