Least Squares

Su Goh

25/10/2020

LM

The LM method in R fits a basic linear model (OLS), regressing y (DV) on x (IV). Here, the model is: $strength_i = \beta_0 + \beta_1 \cdot age_i + \epsilon_i$.

```
data_source <- 'http://www.math.mcgill.ca/yyang/regression/data/2-1-RocketProp.csv'
rocket_data <- read.csv(file=data_source)
names(rocket_data) <- c('i', 'strength', 'age')  #rename the columns
rocket_ols <- lm(strength ~ age, data=rocket_data)

summary(rocket_ols)
```

```
##
## Call:
## lm(formula = strength ~ age, data = rocket_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -215.98  -50.68   28.74   66.61  106.76
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2627.822     44.184   59.48  < 2e-16 ***
## age          -37.154      2.889  -12.86 1.64e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 96.11 on 18 degrees of freedom
## Multiple R-squared:  0.9018, Adjusted R-squared:  0.8964
## F-statistic: 165.4 on 1 and 18 DF,  p-value: 1.643e-10
```

We can call the coefficients with `coefficients(model)`, residuals with `residuals(model)` and so on.

Note that *residual standard error = 96.11 on 18 degrees of freedom* refers to $\hat{\sigma} = \sqrt{\hat{\sigma}^2}$, where: $\hat{\sigma}^2 = \frac{\text{sum of squared residuals}}{n-2} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}$

For the coefficients, std. error refers to $ese(\hat{\beta})$, the estimated standard errors of the estimator.

$$ese(\hat{\beta}_0) = \sqrt{MS_{Res}(\frac{1}{n} + \frac{\bar{x}_1^2}{S_{xx}})}$$

$$ese(\hat{\beta}_1) = \sqrt{\frac{MS_{Res}}{S_{xx}}}$$

The t-value column contains the test statistics $\frac{\hat{\beta}}{ese(\hat{\beta})}$ testing if the coefficient is significantly different from 0 (i.e. $\mathbf{H}_0 : \beta = 0$), and Pr(>|t|) contains the p-value for each test. The asterisks indicate the strength of rejection. For this case, both p-values are significant at $\alpha = 0$, so we can reject both null hypotheses.

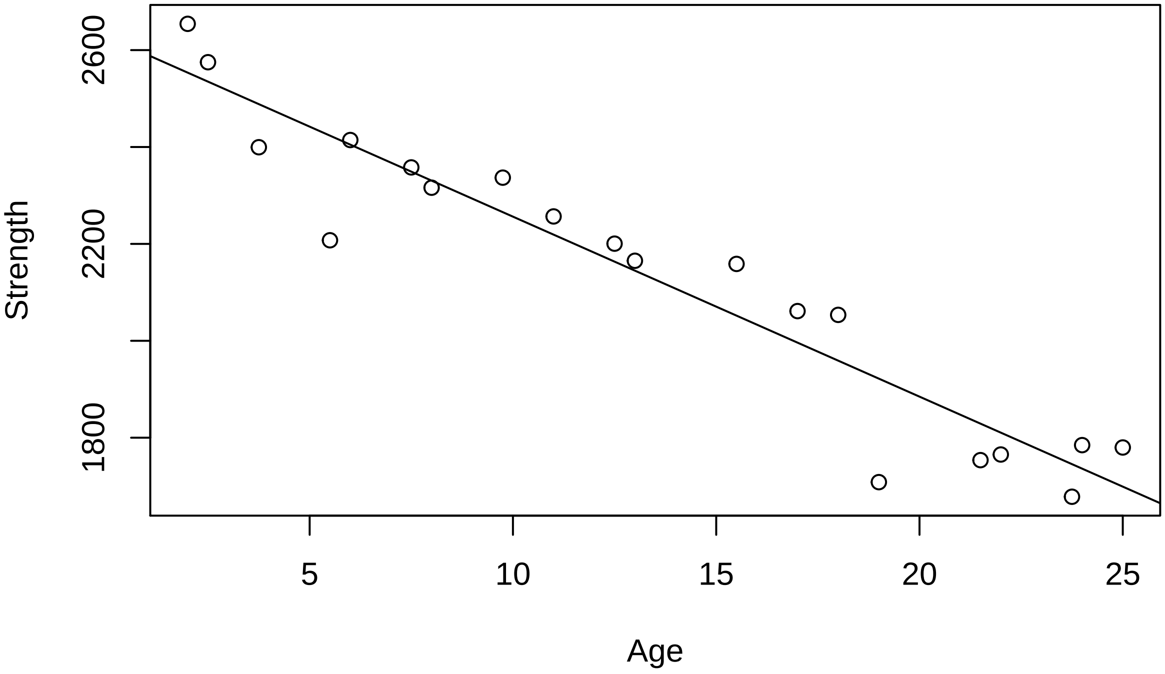
The following gives us predictions for the strength of the rocket, at ages 5 to 10. Note that newdata needs to have the correct column name. If newdata is empty, or if it's malformed, then `predict()` and `fitted()` will give the same result.

```r
test_data = data.frame(age = 5:10)
predict(rocket_ols, newdata = test_data)
```

```
##        1        2        3        4        5        6
## 2442.054 2404.901 2367.747 2330.594 2293.440 2256.286
```

Finally, we plot the results of the data and the regression fit.

```r
plot(strength ~ age, data=rocket_data, xlab='Age', ylab='Strength')
abline(rocket_ols)
```



Matrix form

We can also represent simple linear regression in matrix form: $\begin{pmatrix} Y_1 \\ Y_2 \\ ... \\ Y_n \end{pmatrix} = \begin{pmatrix} 1 & X_{11} \\ 1 & X_{21} \\ ... & \\ 1 & X_{n1} \end{pmatrix} \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ ... \\ \epsilon_n \end{pmatrix}$

Applying OLS to this, we will find that the estimated coefficient is: $\hat{\beta_1} = (X'X)^{-1}X'Y$

We can calculate this directly in R too.

```r
set.seed(1)

# dimensions:
n <- 10  # no. of observations
p <- 2   # no. of params
```

```
xvals <- rnorm(n * p)  #in total have 30 values, drawn from a normal dist
X <- matrix(data=xvals, nrow=n, ncol=p)

Y <- rnorm(n)  # values for Y drawn from a normal dist

beta <- (solve(t(X) %*% X) %*% (t(X) %*% Y))
beta
```

```
##            [,1]
## [1,] -0.7443540
## [2,]  0.3079946
```

Statistical properties of OLS estimators

For the model $Y_i = \beta_0 + \beta_1 X_{i1} + \epsilon_i$, where the SLR assumptions apply, the OLS estimates have properties:

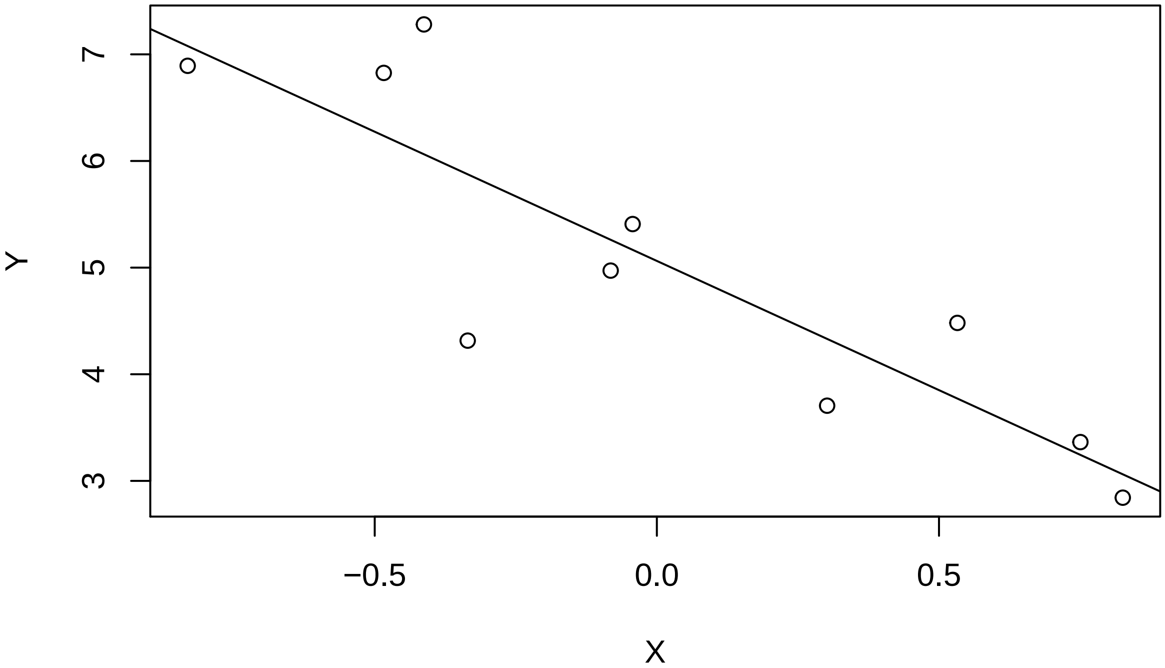
$\mathbb{E}(\hat{\beta}_1) = \beta_1, Var(\hat{\beta}_1) = \frac{\sigma^2}{S_{xx}}$

$\mathbb{E}(\hat{\beta}_0) = \beta_0, Var(\hat{\beta}_0) = \sigma^2(\frac{1}{n} + \frac{\bar{x}_1^2}{S_{xx}})$

Draw n=10 points from $X_i \sim Unif(-1, 1)$ and from $\epsilon_i \sim t_7$. Let $\beta_1 = -2$, $\beta_0 = 5$. Hence the linear model is:

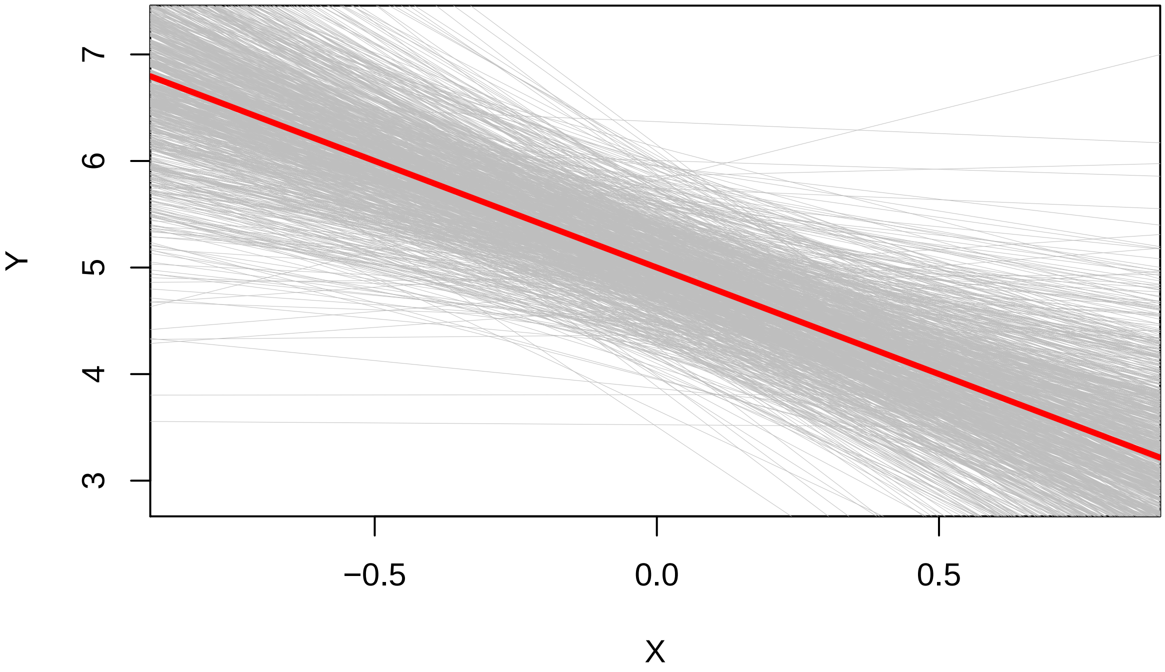
$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

```
beta_0 <- 5
beta_1 <- -2
X <- runif(n=10, -1,1)
e <- rt(n=10, df=7)
Y <- beta_0 + beta_1 * X + e

plot(Y ~ X)
ols <- lm(Y ~ X)
abline(ols)
```



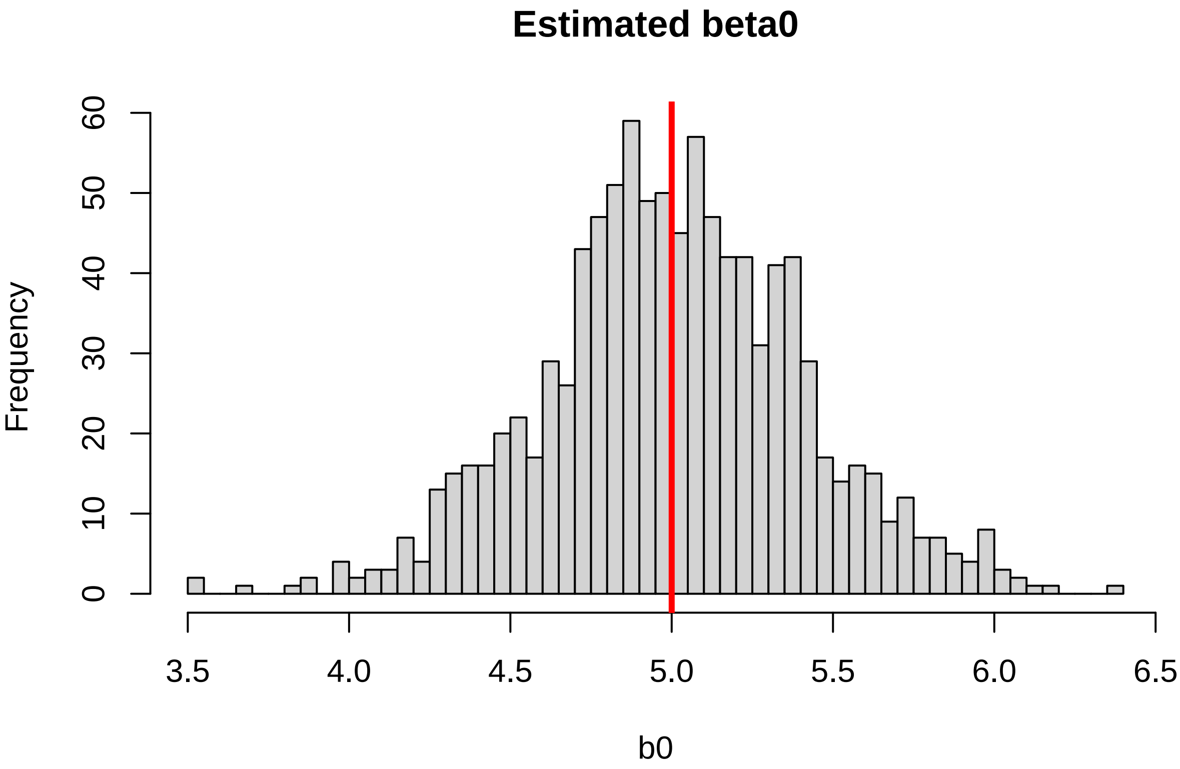
If we repeat the above experiment 1000 times, we draw a different sample each time and thus the sample estimates for the coefficients will be different.

```
nrep <- 1000
b0 <- rep(0, nrep)
b1 <- rep(0, nrep)  #create a null vector

linmod <- function(){
  X <- runif(n=10, -1, 1)
  e <- rt(n=10, df=7)
  Y <- 5 + (-2) * X + e
  return(lm(Y ~ X))
}

plot(NULL, xlim = range(X), ylim=range(Y), xlab='X', ylab='Y')

for(i in 1:nrep){
  model = linmod()
  b0[i] <- model$coef[1]
  b1[i] <- model$coef[2]
  abline(model, col='gray', lw=0.1)
}

abline(a=5, b=-2, lwd=3, col='red')  # true reg line
```



We can also identify the bias $bias = \mathbb{E}(\hat{\beta}_1) - \beta_1$.

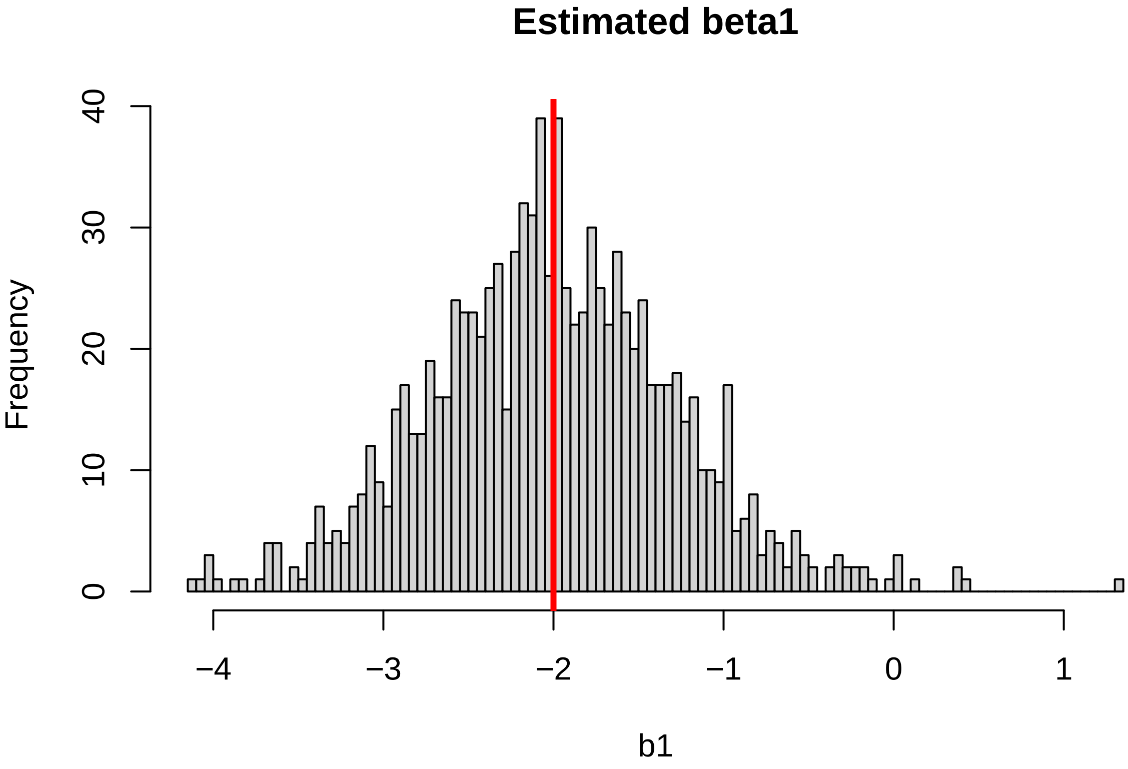
```
hist(b0, main='Estimated beta0', breaks=100)
abline(v=beta_0, col='red', lw=3)
```



```
mean(b0) - beta_0
```

```
## [1] 0.004565176
```

```
hist(b1, main='Estimated beta1', breaks=100)
abline(v=beta_1, col='red', lw=3)
```



```r
mean(b1) - beta_1
```

```
## [1] -0.00327027
```