Predictive Assignment 2

Sreshtha Chatterjee

2026-02-05

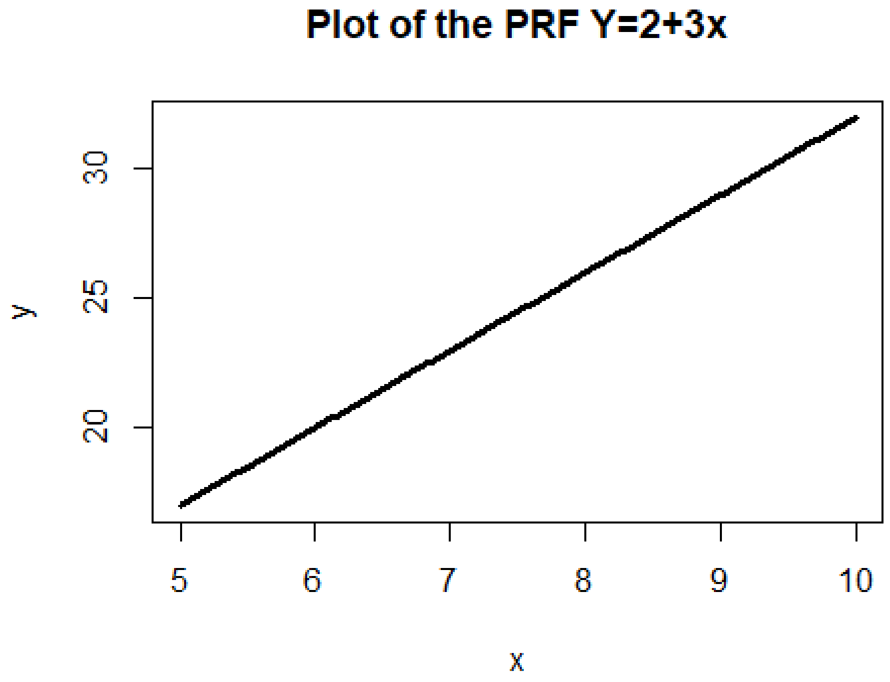
1. Problem to demonstrate that the population regression line is fixed, but least square regression line varies

Model:

$$y = 2 + 3x + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

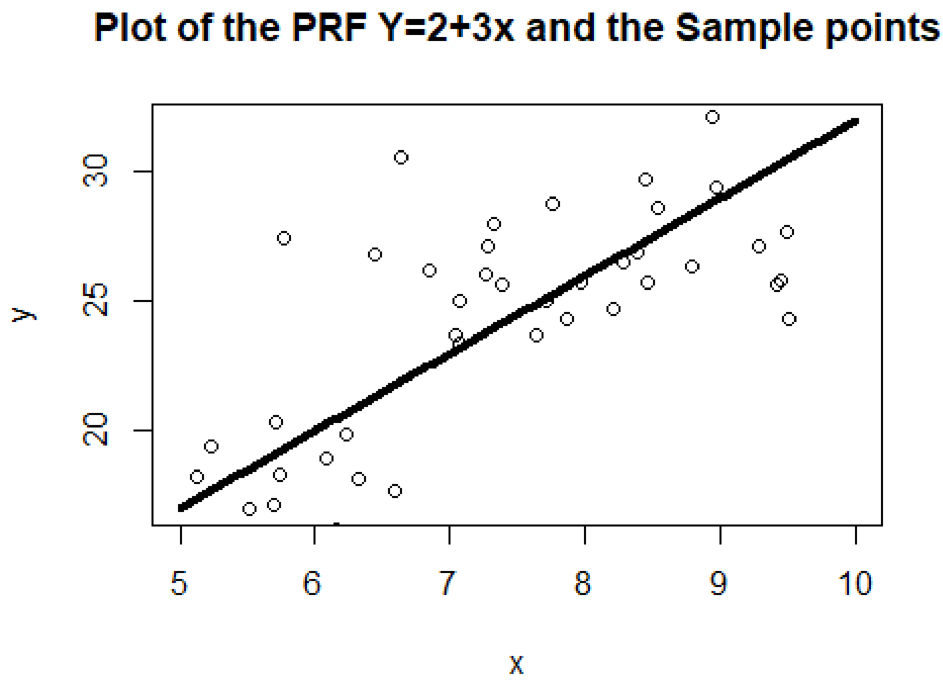
$$y_i \sim U(0,1), \qquad \varepsilon_i \sim \mathcal{N}(0, 4^2), \qquad \forall\, i = 1,2,\dots,n$$

```r
rm(list=ls())

#Step 1: Take 200 equidistant points in the range [5:10]
x=seq(5,10,length.out=200)
y=2+3*x
plot(x,y,type='l',lwd=3,main="Plot of the PRF Y=2+3x")
```



```r
#Step 2: Generate xi from U(5,10) and ei from N(0,16)
n=50
set.seed(123)
xi=runif(n,5,10)
```

```
ei=rnorm(n,0,4)
#SRF y=2+3x+e
yi=2+3*xi+ei
#Overlaying the the points on the PRF
plot(x,y,type='l',lwd=4,main="Plot of the PRF Y=2+3x and the Sample points")
points(xi,yi)
```

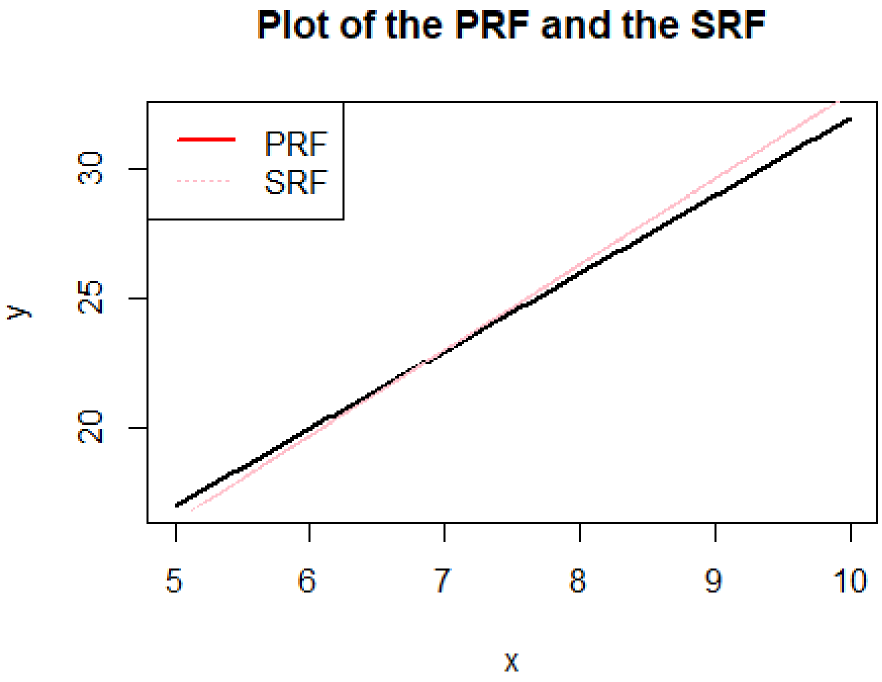


```
#Step 3: Least Square Regression line
model1=lm(yi~xi)
summary(model1)
```

```
##
## Call:
## lm(formula = yi ~ xi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0231 -2.2314 -0.2627  2.1970  8.7445
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.09639    2.82610  -0.034    0.973
## xi           3.30540    0.36519   9.051 5.96e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.761 on 48 degrees of freedom
## Multiple R-squared:  0.6306, Adjusted R-squared:  0.6229
## F-statistic: 81.93 on 1 and 48 DF,  p-value: 5.962e-12
```

$$\hat{y}_i = -0.09639 + 3.30540\,x_i$$

```
#Overlaying the PRF and SRF
y.hat = -0.09639 + 3.30540*xi
plot(x,y,type='l',lwd=2,main="Plot of the PRF and the SRF")
lines(xi,y.hat,type='l',col="pink",lty="dotted")
legend("topleft",col=c("red","pink"),lwd=c(2,1),legend=c("PRF","SRF"),
       lty=c("solid","dotted"))
```



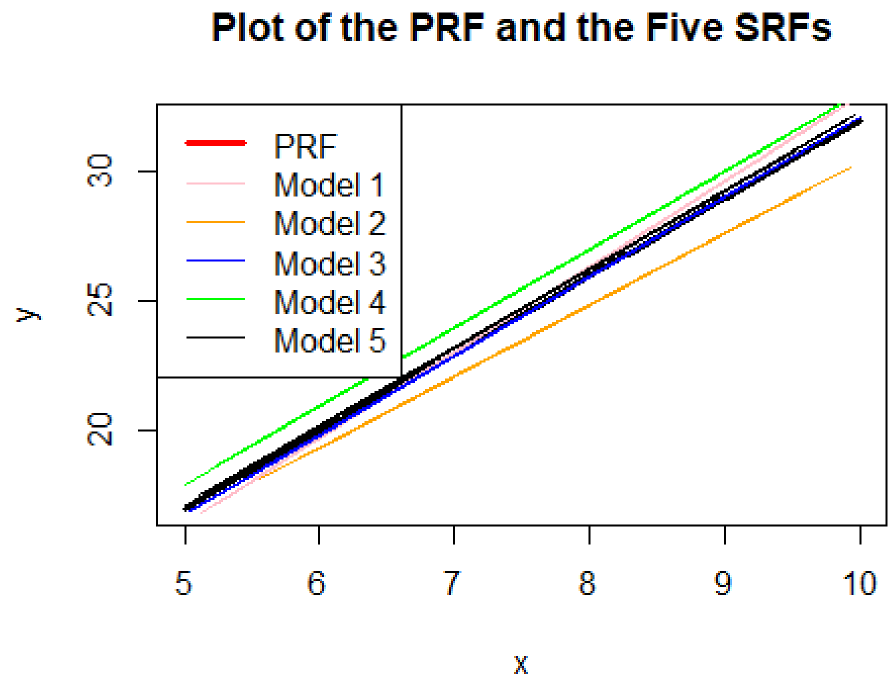
```
#Step 4: Repeat step 2 and 3 for 5 times.
set.seed(123)
x1=runif(n,5,10)
e1=rnorm(n,0,4)
y1=2+3*x1+e1
fit1=lm(y1~x1)
summary(fit1)

##
## Call:
## lm(formula = y1 ~ x1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0231 -2.2314 -0.2627  2.1970  8.7445
```

```
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -0.09639    2.82610  -0.034    0.973    
## x1           3.30540    0.36519   9.051 5.96e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 3.761 on 48 degrees of freedom
## Multiple R-squared:  0.6306, Adjusted R-squared:  0.6229 
## F-statistic: 81.93 on 1 and 48 DF,  p-value: 5.962e-12
```

```
x2=runif(n,5,10)
e2=rnorm(n,0,4)
y2=2+3*x2+e2
fit2=lm(y2~x2)

x3=runif(n,5,10)
e3=rnorm(n,0,4)
y3=2+3*x3+e3
fit3=lm(y3~x3)

x4=runif(n,5,10)
e4=rnorm(n,0,4)
y4=2+3*x4+e4
fit4=lm(y4~x4)

x5=runif(n,5,10)
e5=rnorm(n,0,4)
y5=2+3*x5+e5
fit5=lm(y5~x5)

#Data frame containing the five models' coefficients
coeff=data.frame(coef(fit1),
                 coef(fit2),
                 coef(fit3),
                 coef(fit4),
                 coef(fit5))
coeff
```

```
##              coef.fit1. coef.fit2. coef.fit3. coef.fit4. coef.fit5.
## (Intercept) -0.09638929   2.792188   1.392997   2.823089   2.032506
## x1           3.30539569   2.761042   3.073267   3.023608   3.028097
```

```
#Plot of the PRF and the Five SRFs

plot(x,y,type='l',lwd=3,main="Plot of the PRF and the Five SRFs")
lines(x1,predict(fit1),type='l',col="pink")
lines(x2,predict(fit2),type='l',col="orange")
lines(x3,predict(fit3),type='l',col="blue")
lines(x4,predict(fit4),type='l',col="green")
lines(x5,predict(fit5),type='l',col="black")
legend("topleft",legend=c("PRF","Model 1","Model 2","Model 3",
                          "Model 4","Model 5"),
       col=c("red","pink","orange","blue","green","black"),
       lwd=c(3,1,1,1,1,1))
```



```
#Interpretation: PRF is fixed but SRF varies
```

---

2. Problem to demonstrate that β ˆ0 and β ˆ minimises RSS

Model

$$\hat{y}_i = -0.09639 + 3.30540\, x_i$$

Here

$$\varepsilon_i \sim \mathcal{N}(0,1), \qquad \forall\, i = 1,2,\ldots,n$$

```
rm(list=ls())
#step 1: Generate xi~U(5,10) of size 50, and ei~N(0,1)
#y=2+3x+e
n=50
set.seed(123)
x=runif(n,5,10)
xm=x-mean(x)
```

```r
e=rnorm(n)
y=2+3*xm+e
fit=lm(y~xm)
summary(fit)
```

```
## 
## Call:
## lm(formula = y ~ xm)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -2.25578 -0.55786 -0.06567  0.54926  2.18613 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   2.0562     0.1330   15.46   <2e-16 ***
## xm            3.0764     0.0913   33.70   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.9404 on 48 degrees of freedom
## Multiple R-squared:  0.9594, Adjusted R-squared:  0.9586 
## F-statistic:  1135 on 1 and 48 DF,  p-value: < 2.2e-16
```

$$\hat{\beta}_0 = 2.0562, \qquad \hat{\beta} = 3.0764$$

$$\text{RSS} = (y_1 - \beta_0 - \beta x_1)^2 + (y_2 - \beta_0 - \beta x_2)^2 + \cdots + (y_n - \beta_0 - \beta x_n)^2$$

OR

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \beta_0 - \beta x_i)^2$$

```r
beta0=2.0562
beta=3.0764
#Creating the beta and beta0 grid values
beta0.grid=seq(-1,1,length.out=101)+beta0
beta.grid=seq(-1,1,length.out=101)+beta
#Computing RSS
RSS=c()
for(i in 1:length(beta.grid))
{
  RSS[i]=sum((y-beta0.grid[i]-beta.grid[i]*xm)^2)
}
RSS
```

```
##   [1] 198.52541 192.34441 186.28828 180.35703 174.55065 168.86915 
163.31252
##   [8] 157.88076 152.57388 147.39187 142.33473 137.40247 132.59508 
127.91256
```

```
##  [15] 123.35492 118.92215 114.61425 110.43123 106.37308 102.43981
98.63140
##  [22]  94.94788  91.38922  87.95544  84.64653  81.46249  78.40333
75.46905
##  [29]  72.65963  69.97509  67.41542  64.98063  62.67071  60.48566
58.42549
##  [36]  56.49019  54.67976  52.99421  51.43353  49.99772  48.68679
47.50073
##  [43]  46.43954  45.50323  44.69179  44.00522  43.44353  43.00671
42.69477
##  [50]  42.50769  42.44550  42.50817  42.69572  43.00814  43.44544
44.00761
##  [57]  44.69465  45.50656  46.44335  47.50502  48.69155  50.00296
51.43925
##  [64]  53.00040  54.68643  56.49734  58.43311  60.49376  62.67929
64.98968
##  [71]  67.42495  69.98510  72.67012  75.48001  78.41477  81.47441
84.65892
##  [78]  87.96831  91.40257  94.96170  98.64570 102.45458 106.38833
110.44696
##  [85] 114.63046 118.93883 123.37208 127.93020 132.61319 137.42106
142.35380
##  [92] 147.41141 152.59390 157.90126 163.33349 168.89060 174.57258
180.37943
##  [99] 186.31116 192.36776 198.54924
```

```r
df=data.frame(beta0.grid,beta.grid,RSS)
head(df) #For viewing purpose
```

```
##   beta0.grid beta.grid      RSS
## 1     1.0562    2.0764 198.5254
## 2     1.0762    2.0964 192.3444
## 3     1.0962    2.1164 186.2883
## 4     1.1162    2.1364 180.3570
## 5     1.1362    2.1564 174.5507
## 6     1.1562    2.1764 168.8692
```

```r
df[which.min(RSS),]
```

```
##    beta0.grid beta.grid     RSS
## 51     2.0562    3.0764 42.4455
```

```r
#We can verify the LSE minimises the RSS
```

RSS is minimum at $\beta_0 = 2.0562,\ \beta = 3.0764,$ with RSS $= 42.4455,$ which are the least squares estimates.

3. Problem to demonstrate that least square estimators are unbiased

```
rm(list=ls())
#Step 1: Generate xi~U(5,10) of size 50, and ei~N(0,1)
n=50
set.seed(123)
x=runif(n,0,1)
e=rnorm(n)
y=2+3*x+e
#Step 2: obtain LSE
fit=lm(y~x)
summary(fit)

## 
## Call:
## lm(formula = y ~ x)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -2.25578 -0.55786 -0.06567  0.54926  2.18613 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   1.8576     0.2721   6.827 1.36e-08 ***
## x             3.3817     0.4565   7.408 1.75e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.9404 on 48 degrees of freedom
## Multiple R-squared:  0.5334, Adjusted R-squared:  0.5237 
## F-statistic: 54.88 on 1 and 48 DF,  p-value: 1.745e-09
```

$$\hat{\beta}_0 = 1.8576, \qquad \hat{\beta} = 3.3817$$

```
beta0.hat=1.8576
beta.hat=3.3817
#Repeat R=1000 times
set.seed(123)
R=10000
beta0=numeric(R)
beta=numeric(R)
n=50
for(i in 1:R) {
  x=runif(n,0,1)
  e=rnorm(n,0,1)
  y=2+3*x+e
  fit=lm(y~x)
  beta0[i]=coef(fit)[1]
  beta[i]=coef(fit)[2]
}
```

```
list(Expected_beta0=mean(beta0),Expected_beta=mean(beta))

## $Expected_beta0
## [1] 2.005203
##
## $Expected_beta
## [1] 2.991338

#comment: the long run means of beta0.hat and beta hat converges to true beta
and beta0
```

$$\text{As } n \to \infty, \quad \mathbb{E}(\hat{\beta}_0) \to \beta_0 \quad \text{and} \quad \mathbb{E}(\hat{\beta}) \to \beta$$

```
list(var(beta0),var(beta))

## [[1]]
## [1] 0.08292927
##
## [[2]]
## [1] 0.2495031

#Comment: var approaches 0 as R goes up. Hence LSE is consistent.
```

$$\text{As } n \to \infty, \quad \mathrm{Var}(\hat{\beta}_0) \to 0 \quad \text{and} \quad \mathrm{Var}(\hat{\beta}) \to 0.$$

Hence, by the sufficient condition for consistency, the least squares estimators $\hat{\beta}_0$ and $\hat{\beta}$ are consistent.

---

4. Comparing several simple linear regressions

```
rm(list=ls())
library(MASS)
attach(Boston)
```

$$\begin{aligned}
\text{Model 1:} &\quad \mathrm{medv}_i = \beta_0 + \beta_1\,\mathrm{lstat}_i + \varepsilon_i, \\
\text{Model 2:} &\quad \mathrm{medv}_i = \beta_0 + \beta_1\,\mathrm{crim}_i + \varepsilon_i, \\
\text{Model 3:} &\quad \mathrm{medv}_i = \beta_0 + \beta_1\,\mathrm{nox}_i + \varepsilon_i, \\
\text{Model 4:} &\quad \mathrm{medv}_i = \beta_0 + \beta_1\,\mathrm{black}_i + \varepsilon_i,
\end{aligned}$$

$$\varepsilon_i \sim \mathcal{N}(0, \sigma^2), \qquad i = 1,2,\dots,n$$

β_0 and
β_1 in each case are unknown parameters and are estimated using the method of least squares.

```
fit1=lm(medv~lstat)
fit2=lm(medv~crim)
fit3=lm(medv~nox)
fit4=lm(medv~black)
```

(a) Selecting the predictors one by one, run four separate linear regressions to the data. Present the output in a single table.

```
library(stargazer)
```

```
## Warning: package 'stargazer' was built under R version 4.5.2
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```
stargazer(fit1,fit2,fit3,fit4,type="text",out="f.txt")
```

```
##
## ========================================================================
##                                           Dependent variable:
##                                 -----------------------------------------
##                                                  medv
##                                    (1)        (2)       (3)        (4)
## ------------------------------------------------------------------------
## lstat                          -0.950***
##                                 (0.039)
##
## crim                                      -0.415***
##                                            (0.044)
##
## nox                                                 -33.916***
##                                                      (3.196)
##
## black                                                           0.034***
##                                                                  (0.004)
##
## Constant                       34.554***  24.033*** 41.346***  10.551***
##                                 (0.563)    (0.409)   (1.811)    (1.557)
##
## ------------------------------------------------------------------------
## Observations                       506        506       506        506
## R2                                0.544      0.151     0.183      0.111
## Adjusted R2                       0.543      0.149     0.181      0.109
## Residual Std. Error (df = 504)    6.216      8.484     8.323      8.679
## F Statistic (df = 1; 504)      601.618*** 89.486*** 112.591*** 63.054***
## ========================================================================
## Note:                                        *p<0.1; **p<0.05; ***p<0.01
```

(b) Which model gives the best fit?

Model 1 gives the best fit among the four simple linear regression models since it has the highest coefficient of determination.

The R^2 values for the models are:

$$R_1^2 = 0.544, \quad R_2^2 = 0.151, \quad R_3^2 = 0.183, \quad R_4^2 = 0.111.$$

Since R^2 measures the variability in the dependent variable explained by the predictor, the higher R^2 value for Model 1 indicates that lstat is the best explanatory variable among the four.

Hence, Model 1 provides the best fit to the data.

(c) Comparison and usefulness of predictors:

From the four simple linear regression models with medv as the dependent variable, all predictors are statistically significant at the 1% level, indicating that each variable has a meaningful linear association with median house value.

The coefficient of lstat is -0.950, implying that an increase in the percentage of lower-status population is associated with a substantial decrease in medv. This model also has the highest R^2 value (0.544), suggesting that lstat is the most useful predictor among the four in explaining variability in medv.

The coefficient of crim is -0.415, indicating that higher crime rates are associated with lower median house values. However, its explanatory power is relatively weak, with an R^2 of 0.151, making it less useful compared to lstat.

The coefficient of nox is -33.916, showing a strong negative effect of air pollution on medv. Although the magnitude of the coefficient is large due to the scale of nox, the R^2 value (0.183) suggests moderate usefulness in explaining variation in house values.

The coefficient of black is 0.034, indicating a positive association with medv. Despite being statistically significant, this predictor has the lowest explanatory power with an R^2 of 0.111, making it the least useful among the four.