PREDICTIVE ANALYSIS

SAURABH MISHRA

2026-02-04

PREDICTIVE ANALYTICS

Problem Set 2: Linear Regression

Roll No.: 0709

—

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

Suppose the population regression line is given by Y = 2 + 3x, while the data comes from the model y = 2 + 3x + ε.

Step 1: For x in the range [5,10] graph the population regression line.

Step 2: Generate xi(i = 1, 2, .., n) from Uniform(5, 10) and εi(i = 1, 2, .., n) from N(0, 4^2). Hence, compute y1, y2, .., yn.

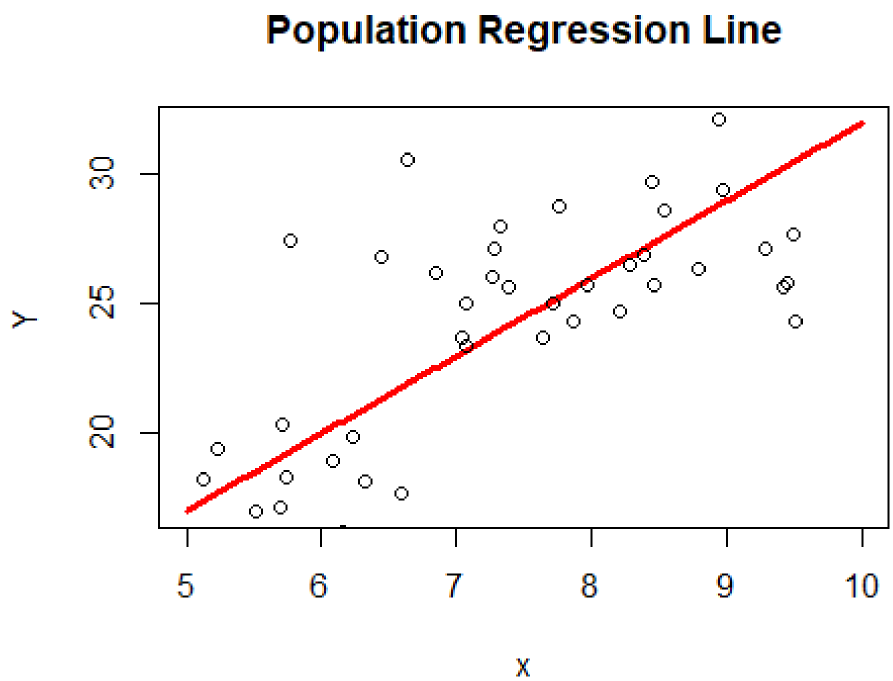
Step 3: On the basis of the data (xi,yi)(i = 1, 2, .., n) generated in Step 2, report the least squares regression line.

Step 4: Repeat steps 2-3 five times. Graph the 5 least squares regression lines over the population regression line obtained in Step 1. Interpret the findings.

Take n = 50. Set the seed as seed=123.

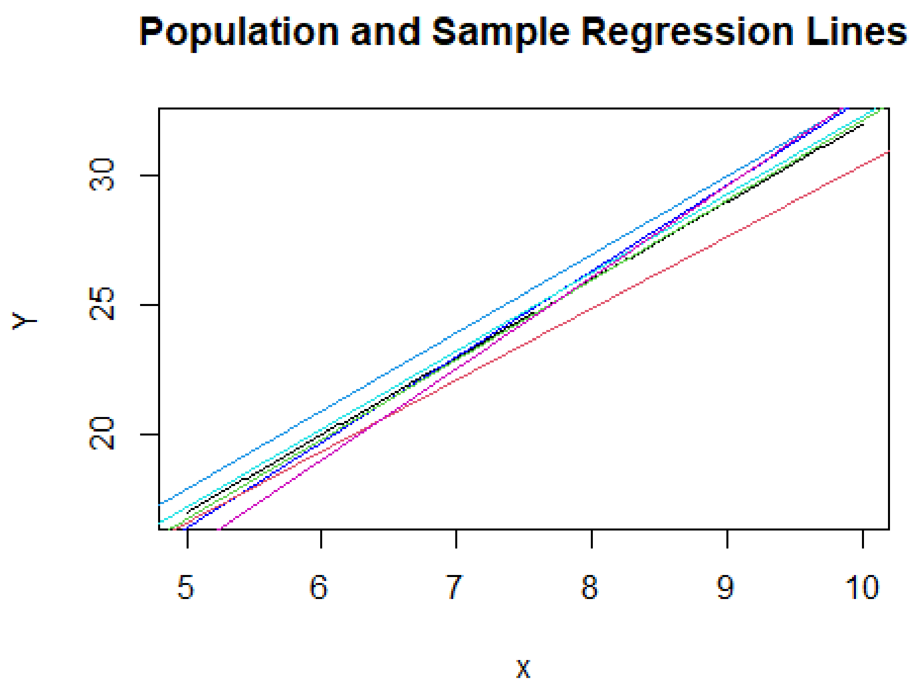
```r
set.seed(123)
n <- 50

x <- seq(5, 10, length.out = 200)
y <- 2 + 3 * x

plot(x, y,
     type = "l",
     lwd = 3,
     col = "red",
     xlab = "x",
     ylab = "Y",
     main = "Population Regression Line")

x1 <- runif(n, 5, 10)
```

```r
e  <- rnorm(n, mean = 0, sd = 4)
y1 <- 2 + 3 * x1 + e

points(x1, y1)
```




```r
lin.reg <- lm(y1 ~ x1)
coef(lin.reg)

## (Intercept)          x1
## -0.09638929  3.30539569

plot(x, y,
     type = "l",
     col = "black",
     xlab = "x",
     ylab = "Y",
     main = "Population and Sample Regression Lines")

abline(lin.reg, col = "blue")

beta.hat <- matrix(NA, nrow = 5, ncol = 2)
colnames(beta.hat) <- c("Intercept", "Slope")

for (i in 1:5) {
  x2  <- runif(n, 5, 10)
  eps <- rnorm(n, 0, 4)
  y2  <- 2 + 3 * x2 + eps
```

```
  lin.reg <- lm(y2 ~ x2)
  beta.hat[i, ] <- coef(lin.reg)

  abline(lin.reg, col = i + 1)
}
```




```
beta.hat

##       Intercept    Slope
## [1,]  2.792188 2.761042
## [2,]  1.392997 3.073267
## [3,]  2.823089 3.023608
## [4,]  2.032506 3.028097
## [5,] -2.107763 3.530691
```

2 Problem to demonstrate that $\hat{\beta}_0$ and $\hat{\beta}$ minimises RSS

Step 1: Generate x_i from Uniform(5, 10) and mean centre the values. Generate ε_i from N(0, 1). Calculate $y_i = 2 + 3x_i + \varepsilon_i$, i = 1,2,.., n. Take n=50 and seed=123.

Step 2: Now imagine that you only have the data on (x_i, y_i), i = 1, 2, .., n, without knowing the mechanism that was used to generate the data in step 1. Assuming a linear regression of the type $y_i = \beta_0 + \beta x_i + \varepsilon_i$, and based on these data (x_i, y_i), i = 1, 2, .., n, obtain the least squares estimates of β_0 and β.

Step 3: Take a large number of grid values of ($\beta 0$, β) that also include the least squares estimates obtained from step 2. Compute the RSS for each parametric choice of ($\beta 0$, β), where RSS = $(y1 - \beta 0 - \beta x1)^2 + (y2 - \beta 0 - \beta x2)^2 + \dots (yn - \beta 0 - \beta xn)^2$. Find out for which combination of ($\beta 0$, β), RSS is minimum.

```r
set.seed(123)
n <- 50

# generate x and mean-center
x_raw <- runif(n, 5, 10)
x <- x_raw - mean(x_raw)

# generate errors
epsilon <- rnorm(n, mean = 0, sd = 1)

# generate y
y <- 2 + 3 * x + epsilon

ols_model <- lm(y ~ x)
beta_hat <- coef(ols_model)

beta_hat

## (Intercept)           x
##    2.056189    3.076349

beta0_grid <- seq(beta_hat[1] - 2, beta_hat[1] + 2, length.out = 100)
beta1_grid <- seq(beta_hat[2] - 2, beta_hat[2] + 2, length.out = 100)

RSS <- matrix(NA, nrow = length(beta0_grid), ncol = length(beta1_grid))


for (i in 1:length(beta0_grid)) {
  for (j in 1:length(beta1_grid)) {

    beta0 <- beta0_grid[i]
    beta1 <- beta1_grid[j]

    RSS[i, j] <- sum((y - beta0 - beta1 * x)^2)
  }
}


min_index <- which(RSS == min(RSS), arr.ind = TRUE)

beta0_min <- beta0_grid[min_index[1]]
beta1_min <- beta1_grid[min_index[2]]
```

```
c(beta0_min, beta1_min)

## [1] 2.035987 3.096551
```

3 Problem to demonstrate that least square estimators are unbiased

Step 1: Generate xi(i = 1, 2, .., n) from Uniform(0, 1), εi(i = 1, 2, .., n) from N(0, 1) and hence generate y using yi = β0 + βxi + εi. (Take β0 = 2, β =3).

Step 2: On the basis of the data (xi, yi)(i = 1, 2, .., n) generated in Step 1, obtain the least square estimates of β0 and β.

Repeat Steps 1-2, R = 1000 times. In each simulation obtain β ˆ0 and β ˆ. Finally, the least-square estimates will be given by the average of these estimated values.

Compare these with the true β0 and β and comment.

Take n = 50 and seed=123.

```
set.seed(123)

n <- 50
R <- 1000

beta0_true <- 2
beta1_true <- 3

# storage for estimates
beta0_hat <- numeric(R)
beta1_hat <- numeric(R)

for (r in 1:R) {

  # generate data
  x <- runif(n, 0, 1)
  epsilon <- rnorm(n, mean = 0, sd = 1)
  y <- beta0_true + beta1_true * x + epsilon

  # OLS estimation
  model <- lm(y ~ x)
  beta0_hat[r] <- coef(model)[1]
  beta1_hat[r] <- coef(model)[2]
}

mean_beta0_hat <- mean(beta0_hat)
mean_beta1_hat <- mean(beta1_hat)

c(mean_beta0_hat, mean_beta1_hat)
```

```
## [1] 2.013053 2.982112

comparison <- data.frame(
  Parameter = c("Intercept (β0)", "Slope (β)"),
  True_Value = c(beta0_true, beta1_true),
  Average_LS_Estimate = c(mean_beta0_hat, mean_beta1_hat)
)

comparison

##        Parameter True_Value Average_LS_Estimate
## 1 Intercept (β0)          2            2.013053
## 2      Slope (β)          3            2.982112
```

4 Comparing several simple linear regressions

Attach "Boston" data from MASS library in R. Select median value of owner- occupied homes, as the response and per capita crime rate, nitrogen oxides concentration, proportion of blacks and percentage of lower status of the popu- lation as predictors.

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

```
library(MASS)

## Warning: package 'MASS' was built under R version 4.5.2

data(Boston)

# Response
y <- Boston$medv

model_crim  <- lm(medv ~ crim, data = Boston)
model_nox   <- lm(medv ~ nox, data = Boston)
model_black <- lm(medv ~ black, data = Boston)
model_lstat <- lm(medv ~ lstat, data = Boston)

results <- data.frame(
  Predictor = c("crim", "nox", "black", "lstat"),
  Intercept = c(coef(model_crim)[1],
                coef(model_nox)[1],
                coef(model_black)[1],
                coef(model_lstat)[1]),
  Slope = c(coef(model_crim)[2],
            coef(model_nox)[2],
            coef(model_black)[2],
            coef(model_lstat)[2]),
  R_squared = c(summary(model_crim)$r.squared,
                summary(model_nox)$r.squared,
                summary(model_black)$r.squared,
                summary(model_lstat)$r.squared),
```

```
  p_value = c(summary(model_crim)$coefficients[2,4],
              summary(model_nox)$coefficients[2,4],
              summary(model_black)$coefficients[2,4],
              summary(model_lstat)$coefficients[2,4])
)

results

##       Predictor Intercept        Slope R_squared      p_value
## crim       crim  24.03311  -0.41519028 0.1507805 1.173987e-19
## nox         nox  41.34587 -33.91605501 0.1826030 7.065042e-24
## black     black  10.55103   0.03359306 0.1111961 1.318113e-14
## lstat     lstat  34.55384  -0.95004935 0.5441463 5.081103e-88
```

(b) Which model gives the best fit?

The lstat model has the highest R^2 (≈ 0.54), indicating that it explains the largest proportion of variation in median house values. This shows that the percentage of lower-status population is the most important single predictor among those considered. Although crime rate and pollution are negatively related to house prices, their explanatory power is much weaker. Hence, lstat provides the best fit among the four models.

(c) Compare the coefficients of the predictors from each model and comment on the usefulness of the predictors.

The coefficients of crim, nox, and lstat are negative, indicating that higher crime rates, greater pollution, and a larger lower-status population are associated with lower housing prices. Among these, lstat has the largest magnitude coefficient, showing the strongest impact on medv. The coefficient of black is positive but relatively small, suggesting a weaker relationship with house values. Overall, lstat is the most useful predictor, while crim and nox are moderately useful and black is the least informative when considered individually.