predictive ps2_735

Srinjana Sen

2026-02-03

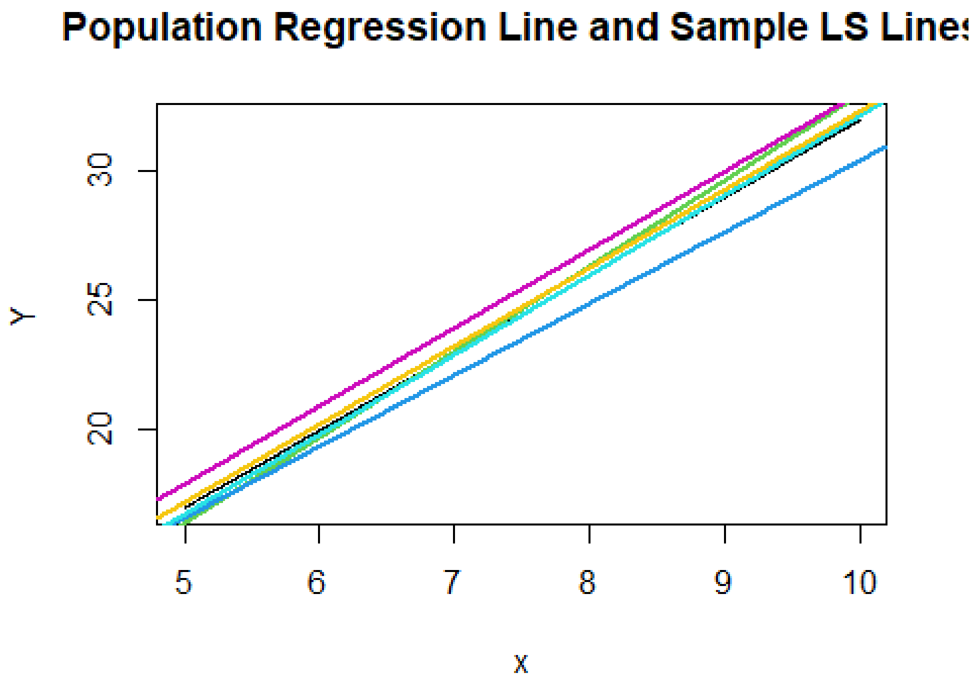
PREDICTIVE ANALYTICS

Problem Set 2: Linear Regression

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
x=seq(5,10,0.001)
Y=2+3*x
plot(x,Y,type="l",main = "Population Regression Line and Sample LS Lines")
set.seed(123)
n=50
b_hat=c()
for(k in 1:5){
  xi=runif(n,5,10)
  ei=rnorm(n,0,4)
  yi=2+3*xi+ei
  fit=lm(yi~xi)
  b_hat=cbind(b_hat,coef(fit))
  abline(fit, col = k+2, lwd = 2)
}
```



```
beta_hat=as.data.frame(b_hat)
beta_hat

##                      V1       V2       V3       V4       V5
## (Intercept) -0.09638929 2.792188 1.392997 2.823089 2.032506
## xi           3.30539569 2.761042 3.073267 3.023608 3.028097
```

2. Problem to demonstrate that beta0_hat and beta_hat minimises RSS

Step 1: Generate xi from Uniform(5, 10) and mean centre the values. Generate ϵi from N(0, 1). Calculate yi = 2 + 3xi + ϵi, i = 1,2,.., n. Take n=50 and seed=123.

```
set.seed(123)
n=50
xi=runif(n,5,10)
xi_new=xi-mean(xi)
ei=rnorm(n,0,1)
yi=2+3*xi+ei
```

Step 2: Now imagine that you only have the data on (xi, yi), 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 yi = β0 + βxi + ϵi, and based on these data (xi, yi), i = 1, 2, .., n, obtain the least squares estimates of β0 and β.

```
model=lm(yi~xi)
b0_hat=coef(model)[1]
```

```
b1_hat=coef(model)[2]
b0_hat

## (Intercept)
##    1.475903

b1_hat

##       xi
## 3.076349
```

Step 3: Take a large number of grid values of (β0, β) that also include the least squares estimates obtained from step 2. Compute the RSS for each parametric choice of (β0, β), where RSS = (y1 − β0 − βx1)^2 + (y2 − β0 − βx2)^2 + ...(yn −β0 − βxn)^2. Find out for which combination of (β0, β), RSS is minimum.

```
b0_grid=seq(b0_hat - 2, b0_hat + 2, length.out = 100)
b_grid=seq(b1_hat  - 2, b1_hat  + 2, length.out = 100)

rss_cal=function(b0, b1, x, y) {
  y_pred=b0 + b1 * x
  sum((y - y_pred)^2)
}

rss_val=matrix(NA, nrow = length(b0_grid), ncol = length(b_grid))

rss_min=Inf
best_b0=NA
best_b=NA

for (i in 1:length(b0_grid)) {
  for (j in 1:length(b_grid)) {
    rss_value=rss_cal(b0_grid[i], b_grid[j], xi, yi)
    rss_val[i, j]=rss_value

    if (rss_value < rss_min) {
      rss_min=rss_value
      best_b0=b0_grid[i]
      best_b=b_grid[j]
    }
  }
}

{
print("Grid Minimum RSS is at:")
cat("Beta0:", best_b0, "\n")
cat("Beta:", best_b, "\n")
cat("Minimum RSS:", rss_min, "\n")
}
```

```
## [1] "Grid Minimum RSS is at:"
## Beta0: 1.334489
## Beta: 3.096551
## Minimum RSS: 42.49615
```

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 β0_hat and β_hat. 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.

```
n=50
set.seed(123)
b0=numeric(1000)
b1=numeric(1000)
for(k in 1:1000){
xi=runif(n,0,1)
ei=rnorm(n,0,1)
yi=2+3*xi+ei
model=lm(yi~xi)
b0[k]=coef(model)[1]
b1[k]=coef(model)[2]
}
b0mean=mean(b0)
b1mean=mean(b1)
b0mean

## [1] 2.013053

b1mean

## [1] 2.982112

b0var=var(b0)
b1var=var(b1)
b0var

## [1] 0.07969639

b1var

## [1] 0.2360544
```

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

```r
library(MASS)
data("Boston")
preds = c("crim", "nox", "black", "lstat")
output = data.frame()

for(var in preds){
  formula = as.formula(paste("medv ~", var))
  lin_model = lm(formula, data = Boston)
  summary = summary(lin_model)
  summary

  output = rbind(output, data.frame(
    Pred = var,
    Intercept = round(coef(model)[1], 4),
    Coeff = round(coef(model)[2], 4),
    StdE = round(summary$coefficients[2, 2], 4),
    RSq = round(summary$r.squared, 4),
    RSE = round(summary$sigma, 4)
  ))
}

output
```

```
##                  Pred Intercept  Coeff   StdE    RSq    RSE
## (Intercept)   crim    2.1648 2.5832 0.0439 0.1508 8.4838
## (Intercept)1   nox    2.1648 2.5832 3.1963 0.1826 8.3233
## (Intercept)2 black    2.1648 2.5832 0.0042 0.1112 8.6793
## (Intercept)3 lstat    2.1648 2.5832 0.0387 0.5441 6.2158
```

(b) Which model gives the best fit?
From the table, it is clear that the regression model with lstat provides the best overall fit.

- *R-Squared:* The lstat model has the highest value of 0.5441, meaning it explains about 54% of the variation in median house prices.

- The next best predictor, nox, has an R-squared of 0.1826 (around 18%).

- *Residual Standard Error (RSE):* The lstat model also produces the smallest prediction error, with an RSE of 6.2158, making it the most accurate among the models compared.

(c) Compare the coefficients of the predictors from each model and comment on the usefulness of the predictors.
we interpret the regression coefficients in terms of their importance and direction:

- *lstat (Coef: -0.95):* This is the strongest predictor. The negative coefficient indicates that as the proportion of the lower-status population increases, median home values decrease significantly.
- *nox (Coef: -33.9161):* The coefficient is also negative, suggesting that higher levels of air pollution are associated with lower housing prices.
- *crim (Coef: -0.4152):* Crime rate shows a negative effect on home values, but its explanatory strength is much weaker compared to lstat.
- *black (Coef: 0.0336):* Although the coefficient is positive and statistically meaningful, this variable has the lowest explanatory power among the four predictors, making it the least useful single predictor in this comparison.