qbs121_hw1_gibran

Gibran Erlangga

1/11/2022

Homework 1 - Linear Models

Question 1 (bonus)

a. Show that the sample mean $\sum_{i=1}^{n} x_i/n$ minimizes the average squared distance, $\sum(x_i - \mu)^2$.

Let's say we have $s(a) = \sum_{i=1}^{n}(x_i - a)^2$

$$\begin{aligned}\sum_{i=1}^{n}(x_i - a)^2 &= \sum_{i=1}^{n}(x_i^2 - 2x_i a + a^2) \\ &= \sum_{i=1}^{n} x_i^2 + 2a\sum_{i=1}^{n} x_i + na^2\end{aligned}$$

To minimize s(a), take first derivative. We get $s'(a) = -2\sum_{i=1}^{n} x_i + 2na$, which will be zero when $\sum_{i=1}^{n} x_i/n$.

b. Show that the median minimizes the average distance, $\sum |x_i - \mu|$

Assuming that the set S has n elements and $s_1 < s_2 < \ldots < s_n$. If x $< s_1$, it becomes

$$\begin{aligned}f(x) &= \sum_{s\in S} |s - x| \\ &= \sum_{s\in S}(s - x) \\ &= \sum_{k=1}^{n}(s_k - x)\end{aligned}$$

Suppose that $S_k \leq x \leq x + d \leq s_{k+1}$. Then,

$$
\begin{aligned}
f(x+d) &= \sum_{i=1}^{k}(x+d-s_i) + \sum_{i=1}^{n}(s_i-(x+d)) \\
&= dk + \sum_{i=1}^{k}(x-si) - d(n-k) + \sum_{i=k+1}^{n}(s+i-x) \\
&= d(2k-n) + \sum_{i=1}^{k}(x-si) + \sum_{i=k+1}^{n}(s+i-x) \\
&= d(2k-n) + f(x) \\
f(x+d) - f(x) &= d(2k-n)
\end{aligned}
$$

So, $f(x)$ will be negative if $2k < n$, 0 if $2k = n$ and $2k > n$. Therefore, $f(x)$ is minimal when x is the median of S.

Question 4 (bonus)

Given a dependent variable Y and features $X_1, \ldots, X_k$ find the linear combination of the features that maximizes the correlation with Y.

Question 5 (bonus)

How does R^2 change if (a) the dependent variable Y is rescaled, or (b) a new predictor $X_3 = aX_1 + bX_2$ is added to the linear model $Y = \beta_0 + \beta_1 X_2 + \beta_2 X_2 + \epsilon$.

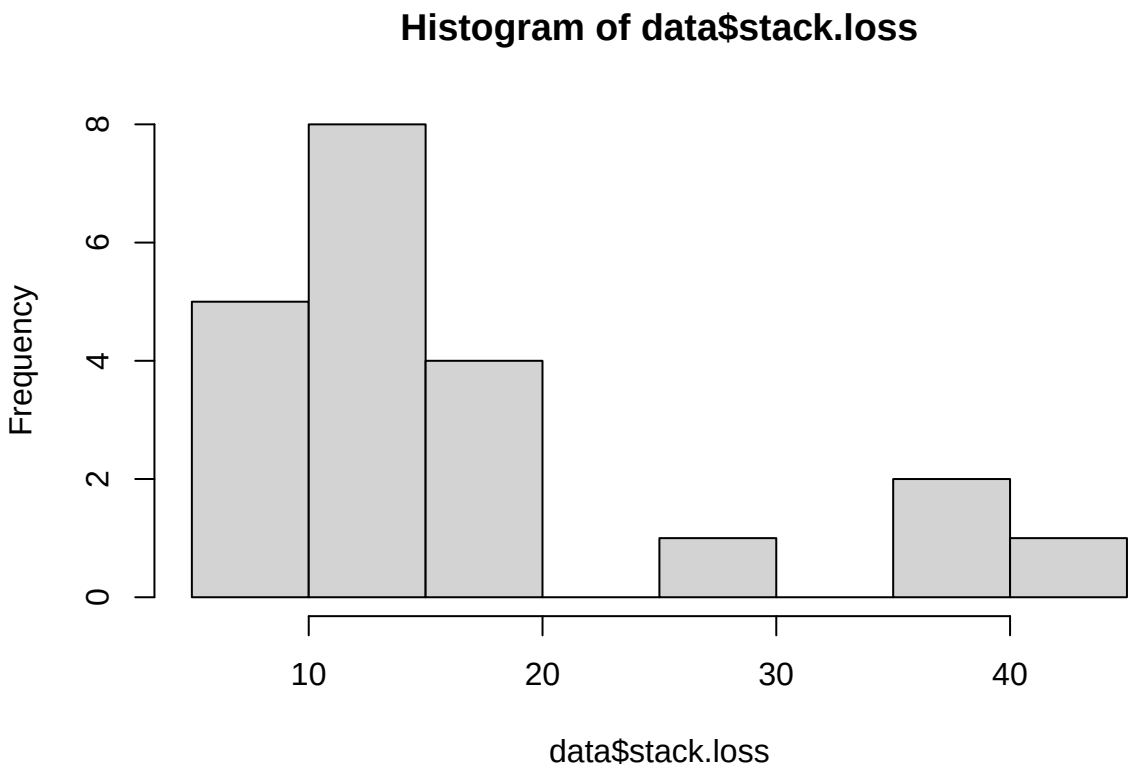
2.2

```
data <- stackloss
head(data, 2)
```

```
##   Air.Flow Water.Temp Acid.Conc. stack.loss
## 1       80         27         89         42
## 2       80         27         88         37
```

1.Comment on the distribution of stack.loss.

```
hist(data$stack.loss)
```



The distribution is skewed to the right, where the majority of values get together on the left and long tail to the right.

2. Regress yield on Air.Flow, Water.Temp and Acid.Conc., one at a time (univariable models).

```
summary(lm(stack.loss ~ Air.Flow, data))
```

```
##
## Call:
## lm(formula = stack.loss ~ Air.Flow, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.2896  -1.1272  -0.0459   1.1166   8.8728
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -44.13202    6.10586  -7.228 7.31e-07 ***
## Air.Flow      1.02031    0.09995  10.208 3.77e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.098 on 19 degrees of freedom
## Multiple R-squared:  0.8458, Adjusted R-squared:  0.8377
```

```
## F-statistic: 104.2 on 1 and 19 DF,  p-value: 3.774e-09
```

```
summary(lm(stack.loss ~ Water.Temp, data))
```

```
##
## Call:
## lm(formula = stack.loss ~ Water.Temp, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8904 -3.6206  0.3794  2.8398  8.4747
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -41.9109     7.6056  -5.511 2.58e-05 ***
## Water.Temp    2.8174     0.3567   7.898 2.03e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.043 on 19 degrees of freedom
## Multiple R-squared:  0.7665, Adjusted R-squared:  0.7542
## F-statistic: 62.37 on 1 and 19 DF,  p-value: 2.028e-07
```

```
summary(lm(stack.loss ~ Acid.Conc., data))
```

```
##
## Call:
## lm(formula = stack.loss ~ Acid.Conc., data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.584  -5.584  -3.066   1.247  22.416
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -47.9632    34.5044  -1.390   0.1806
## Acid.Conc.    0.7590     0.3992   1.901   0.0725 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.565 on 19 degrees of freedom
## Multiple R-squared:  0.1599, Adjusted R-squared:  0.1156
## F-statistic: 3.615 on 1 and 19 DF,  p-value: 0.07252
```

3. Calculate a Pearson correlation of Air.Flow with stackloss and compare this result to the univariable regression of stackloss on Air.Flow above.

```
cor(data$Air.Flow, data$stack.loss, method = c("pearson"))
```

```
## [1] 0.9196635
```

4. Run a multivariable model of stackloss on all three variables. Interpret the coefficients.

```
lm_all <- lm(stack.loss ~ Air.Flow + Water.Temp + Acid.Conc., data)
summary(lm_all)
```

```
##
## Call:
## lm(formula = stack.loss ~ Air.Flow + Water.Temp + Acid.Conc.,
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.2377 -1.7117 -0.4551  2.3614  5.6978
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -39.9197    11.8960  -3.356  0.00375 **
## Air.Flow      0.7156     0.1349   5.307  5.8e-05 ***
## Water.Temp    1.2953     0.3680   3.520  0.00263 **
## Acid.Conc.   -0.1521     0.1563  -0.973  0.34405
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.243 on 17 degrees of freedom
## Multiple R-squared:  0.9136, Adjusted R-squared:  0.8983
## F-statistic:  59.9 on 3 and 17 DF,  p-value: 3.016e-09
```

Intercept: -39.9197, when other variables equal to 0, then the estimated value of stack.loss is the intercept.

slope for Air.Flow: 0.71 -> for every unit increase in Air.Flow, there is a 0.71 increase in the value of stack.loss.

slope for Water.Temp: 1.2953 -> for every unit increase in Water.Temp, there is a 1.2953 increase in the value of stack.loss.

slope for Acid.Conc.: -0.1521 -> for every unit increase in Acid.Conc., there is a 0.1521 decrease in the value of stack.loss.

5. This part illustrates the explicit formula for least squares estimation.

```
design_matrix <- model.matrix(~Air.Flow + Water.Temp + Acid.Conc., data)
design_matrix
```

a. Create the "design matrix" or "model matrix" corresponding to the main effects for Air.Flow, Water.Temp and Acid.Conc. e.g X <- cbind(1, Air.Flow,Water.Temp,Acid.Conc)

```
##    (Intercept) Air.Flow Water.Temp Acid.Conc.
## 1            1       80         27         89
## 2            1       80         27         88
## 3            1       75         25         90
```

```
## 4            1       62         24         87
## 5            1       62         22         87
## 6            1       62         23         87
## 7            1       62         24         93
## 8            1       62         24         93
## 9            1       58         23         87
## 10           1       58         18         80
## 11           1       58         18         89
## 12           1       58         17         88
## 13           1       58         18         82
## 14           1       58         19         93
## 15           1       50         18         89
## 16           1       50         18         86
## 17           1       50         19         72
## 18           1       50         19         79
## 19           1       50         20         80
## 20           1       56         20         82
## 21           1       70         20         91
## attr(,"assign")
## [1] 0 1 2 3
```

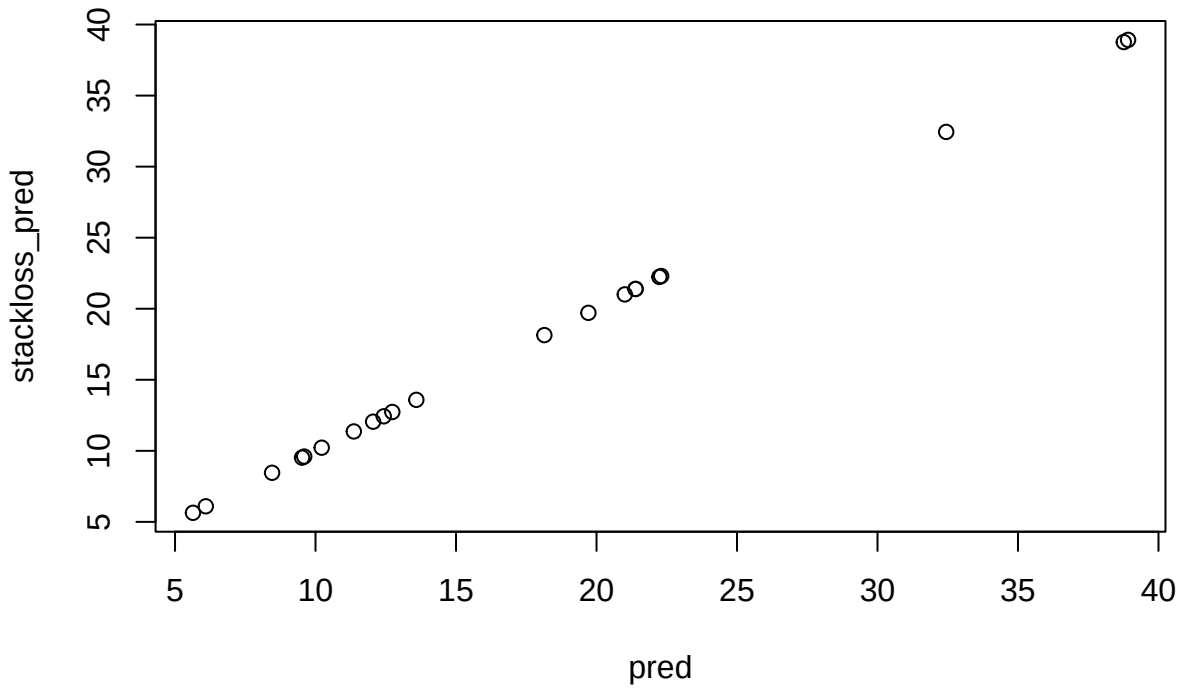
```
pred <- design_matrix %*% solve(t(design_matrix) %*% design_matrix) %*% t(design_matrix) %*% stack.loss
pred
```

b. Calculate pred <- X %% *solve(t(X) %%* X) %% *t(X) %* % yield

```
##         [,1]
## 1  38.765363
## 2  38.917485
## 3  32.444467
## 4  22.302226
## 5  19.711654
## 6  21.006940
## 7  21.389491
## 8  21.389491
## 9  18.144379
## 10 12.732806
## 11 11.363703
## 12 10.220540
## 13 12.428561
## 14 12.050499
## 15  5.638582
## 16  6.094949
## 17  9.519951
## 18  8.455093
## 19  9.598257
## 20 13.587853
## 21 22.237713
```

```r
# compare pred vs lm_all prediction
stackloss_pred <- predict.lm(lm_all, data[c("Air.Flow", "Water.Temp", "Acid.Conc.")])

plot(pred, stackloss_pred)
```

c. Compare pred with the predicted values when you run lm(yield ~ Air.Flow + Water.Temp + Acid.Conc)



4. Simulations

Question 1

```r
#reference: https://blog.revolutionanalytics.com/2016/08/simulating-form-the-bivariate-normal-distribut
library(MASS)

n <- 500
mean <- c(1, 2)
std <- c(3, 4)
rho <- 0.5
cov <- matrix(c(std[1]^2, std[1]*std[2]*rho, std[1]*std[2]*rho, std[2]^2), ncol=2)

data <- mvrnorm(n=n, mu = mean, Sigma = cov)
```

```r
x1 <- data[,1]
x2 <- data[,2]
```

a. Simulate two variables, X_1 and X_2, whose joint distribution is the bivariate normal with means of 1 and 2 respectively, standard deviations of 3 and 4, respectively and correlation of 0.5. Use a sample size of 500.

```r
cor(x1, x2, method = "pearson")
```

b. Calculate the Pearson correlation of the two simulated variables.

```
## [1] 0.5147668
```

```r
model1 <- lm(x1 ~ x2)
summary(model1)
```

c. Calculate the R^2 when X_2 is regressed on X_1.

```
##
## Call:
## lm(formula = x1 ~ x2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.2121 -1.8171 -0.2276  1.6839  9.0447
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.18993    0.13103    1.45    0.148
## x2           0.40219    0.03002   13.40   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.636 on 498 degrees of freedom
## Multiple R-squared:  0.265,  Adjusted R-squared:  0.2635
## F-statistic: 179.5 on 1 and 498 DF,  p-value: < 2.2e-16
```

```r
model2 <- lm(x2 ~ x1)
summary(model2)
```

d. Calculate the R^2 when X_1 is regressed on X_2.

```
##
## Call:
## lm(formula = x2 ~ x1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.1925  -2.0912  -0.0654   2.0932   9.3764
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.27482    0.15805   8.066 5.46e-15 ***
## x1           0.65885    0.04917  13.399  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.374 on 498 degrees of freedom
## Multiple R-squared:  0.265,  Adjusted R-squared:  0.2635
## F-statistic: 179.5 on 1 and 498 DF,  p-value: < 2.2e-16
```

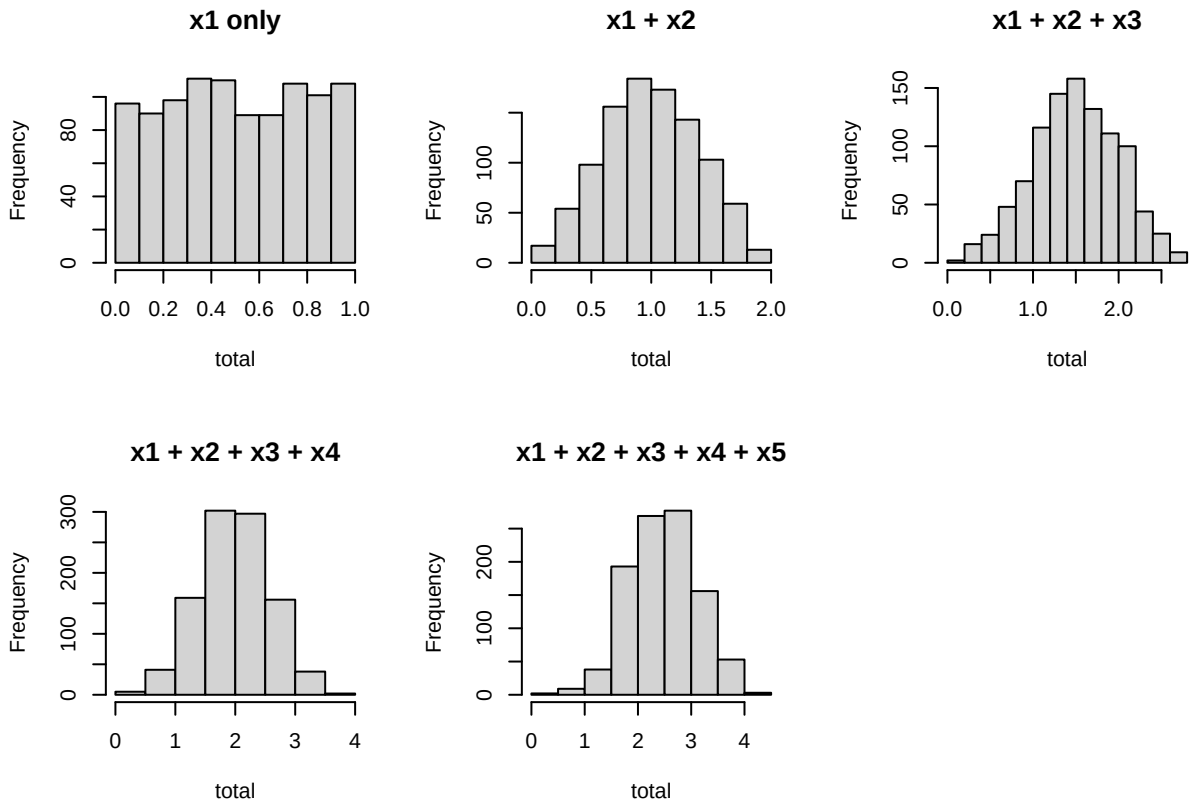
e. Comment on the values reported for parts b,c and d.

Correlation score explains the strength of a relationship between two variables (dependent and independent variable, in this case), while R^2 explains to what extent the variance of one variable explains the variance of another variable. The pearson correlation score for x1 and x2 is 0.53, while the R^2 of model 1 (X_2 is regressed on X_1) is 0.2772, which happened to be also the value for R^2 of model 2 (X_1 is regressed on X_2). For both model 1 and model 2, it explains ~28% variation of the dependent variable.

```
n = 1000

x1 <- runif(n)
x2 <- runif(n)
x3 <- runif(n)
x4 <- runif(n)
x5 <- runif(n)

sum1 <- x1
sum2 <- x1 + x2
sum3 <- x1 + x2 + x3
sum4 <- x1 + x2 + x3 + x4
sum5 <- x1 + x2 + x3 + x4 + x5

par(mfrow=c(2, 3))
hist(sum1, main="x1 only", xlab="total")
hist(sum2, main="x1 + x2", xlab="total")
hist(sum3, main="x1 + x2 + x3", xlab="total")
hist(sum4, main="x1 + x2 + x3 + x4", xlab="total")
hist(sum5, main="x1 + x2 + x3 + x4 + x5", xlab="total")
```

4. The uniform distribution is symmetric but not bell-shaped. Generate 5 random variables, $X_1, X_2, \ldots, X_5$, that are uniformly distributed (U[0,1]) and independent. Created histograms of

$X_1, X_1+X_2, \ldots, X_1+\cdots+X_5$ **. Comment on the shape as you add more of these random variables**



together.

From the graphs above we can observe that as we combine more data points into one value, the distribution of the summation tends to form a normal distribution, although the individual distribution is a uniform distribution and not a normal one.