Lab 4

Burhan Ahmed Hanif

11:59PM March 9, 2019

Note: the content of this lab is on the midterm exam (March 5) even though the lab itself is due after the midterm exam.

We now move on to simple linear modeling using the ordinary least squares algorithm.

Let's quickly recreate the sample data set from practice lecture 7:

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

```
n = 20
x = runif(n)
beta_0 = 3
beta_1 = -2
y = beta_0 + beta_1 * x + rnorm(n, mean = 0, sd = 0.33)
```

Solve for the least squares line by computing b_0 and b_1 *without* using the functions `mean`, `cor`, `cov`, `var`, `sd` but instead computing it from the x and y quantities manually using base function such as `sum` and other basic operators. See the class notes.

```
y_bar = sum(y)/n
x_bar = sum(x)/n
b_1 = (sum(x*y)-(n*x_bar*y_bar))/(sum(x^2)-n*(x_bar)^2)
b_0 = y_bar
b_0
```

```
## [1] 1.691759
```

```
b_1
```

```
## [1] -1.670815
```

Verify your computations are correct using the `lm` function in R:

```
lm_mod = lm(y~x)
b_vec = coef(lm_mod)
b_vec[2]
```

```
##         x
## -1.670815
```

```
b_1
```

```
## [1] -1.670815
```

```
pacman::p_load(testthat)
#expect_equal(b_0, as.numeric(b_vec[1]), tol = 1e-4)
expect_equal(b_1, as.numeric(b_vec[2]), tol = 1e-4)
```

6. We are now going to repeat one of the first linear model building exercises in history — that of Sir Francis Galton in 1886. First load up package `HistData`.

```
#TO-DO
#pacman :: p_load("HistData")
library("HistData")
```

```
## Warning: package 'HistData' was built under R version 3.5.3
```

In it, there is a dataset called `Galton`. Load it up.

```
data(Galton)
```

You now should have a data frame in your workspace called `Galton`. Summarize this data frame and write a few sentences about what you see. Make sure you report n, p and a bit about what the columns represent and how the data was measured. See the help file `?Galton`.

```
?Galton
```

```
## starting httpd help server ... done
```

TO-DO

Find the average height (include both parents and children in this computation).

```
avg_height = mean(mean(Galton$parent) + mean(Galton$child))
```

If you were to use the null model, what would the RMSE be of this model be?

```
n = nrow(Galton)
e = ((Galton$parent+Galton$child) - avg_height)
SSE= sum(e^2)
MSE = SSE/n-2
RMSE = sqrt(MSE)
```

Note that in Math 241 you learned that the sample average is an estimate of the "mean", the population expected value of height. We will call the average the "mean" going forward since it is probably correct to the nearest tenth of an inch with this amount of data.

Run a linear model attempting to explain the childrens' height using the parents' height. Use `lm` and use the R formula notation. Compute and report b_0, b_1, RMSE and R^2. Use the correct units to report these quantities.

```
lin_mod_Galton = lm(Galton$child~Galton$parent)
b_0 = coef(lin_mod_Galton)[1]
b_1 = coef(lin_mod_Galton)[2]
r_sqrd = summary(lin_mod_Galton)$r.squared
lin_mod_Galton_rmse = summary(lin_mod_Galton)$sigma
b_0
```

```
## (Intercept)
##    23.94153
```

```
b_1
```

```
## Galton$parent
##     0.6462906
```

```
r_sqrd
```

```
## [1] 0.2104629
```

```
lin_mod_Galton_rmse
```

```
## [1] 2.238547
```

Interpret all four quantities: b_0, b_1, RMSE and R^2.

b__0 : This coefficent that determines the intercept for the linear model and if the parent height were 0

b__1 : This coefficent determines the slope. in this case for the height of the children increase per unit increase in parent height.

RMSE ; 95% of the data is within 4.5 units of the average

R^2 ; data is scattered all over the line and because the value is small and many of the y's = heights were the same SST is close to SSE

How good is this model? How well does it predict? Discuss. #it depends how we view each metric of R^2 and RMSE for the R^2 being a low value means that the null variance is not explained well in our model within the range of data so it does not predict the variance well. While the RMSE is within 4.5 units that is really good.

TO-DO

It is reasonable to assume that parents and their children have the same height? Explain why this is reasonable using basic biology and common sense.

they have the same height because of genetics and DNA inherited by the parents, would cause similiar heights.

TO-DO

If they were to have the same height and any differences were just random noise with expectation 0, what would the values of β_0 and β_1 be?

beta__0 = 0 becasue

beta__1 = 1 becasue they would be the same height

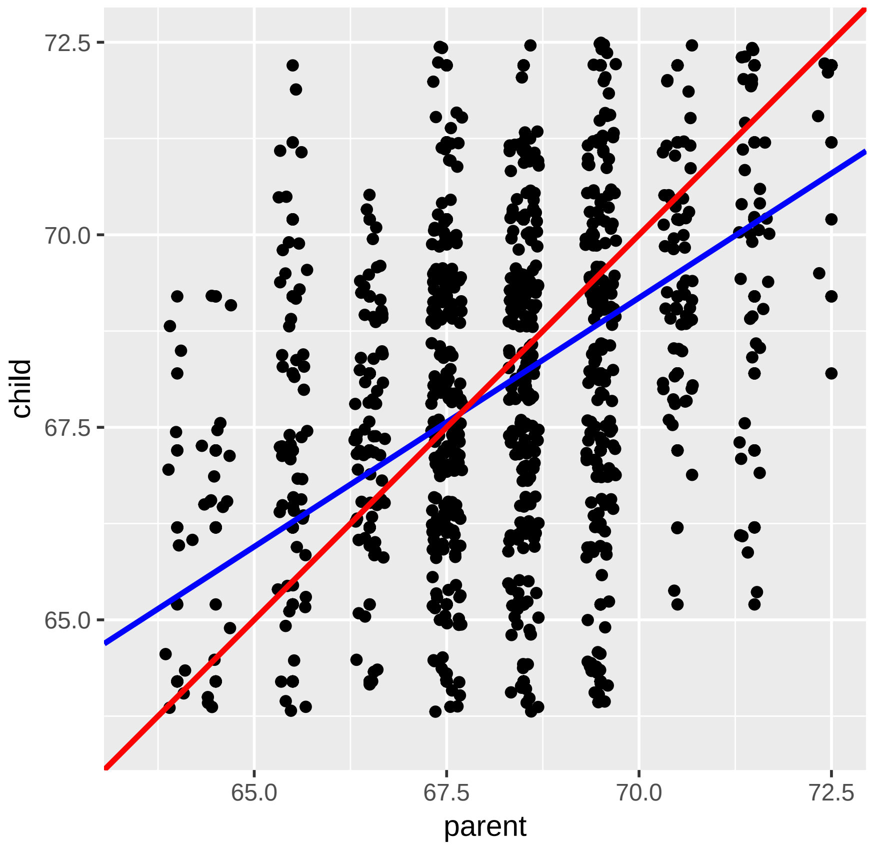
TO-DO

Let's plot (a) the data in $\mathbb{D}$ as black dots, (b) your least squares line defined by b_0 and b_1 in blue, (c) the theoretical line β_0 and β_1 if the parent-child height equality held in red and (d) the mean height in green.

```r
pacman::p_load(ggplot2)
ggplot(Galton, aes(x = parent, y = child)) +
  geom_point() +
  geom_jitter() +
  geom_abline(intercept = b_0, slope = b_1, color = "blue", size = 1) +
  geom_abline(intercept = 0, slope = 1, color = "red", size = 1) +
  geom_abline(intercept = avg_height, slope = 0, color = "darkgreen", size = 1) +
  xlim(63.5, 72.5) +
  ylim(63.5, 72.5) +
  coord_equal(ratio = 1)
```

```
## Warning: Removed 76 rows containing missing values (geom_point).
```

```
## Warning: Removed 89 rows containing missing values (geom_point).
```



Fill in the following sentence:

TO-DO: Children of short parents became longer on average and children of tall parents became shorter on average.

Why did Galton call it "Regression towards mediocrity in hereditary stature" which was later shortened to "regression to the mean"? # over time regressing over the mean at both ends of the interval would sugeest that over each passing generation the taller and shorter family will have children that would be the average height

Why should this effect be real?

the law of large numbers states as the sample size increases , the probability would be expected value

You now have unlocked the mystery. Why is it that when modeling with y continuous, everyone calls it "regression"? Write a better, more descriptive and appropriate name for building predictive models with y continuous.

TO-DO # it is called regression because we are regressing over the features. linear regression is how the average value changes of the dependent variable given that the independentdent variables remain fixed. "dependent variable analysis".

Create a dataset $\mathbb{D}$ which we call `Xy` such that the linear model as R^2 about 50% and RMSE approximately 1.

```
x = c(1 , 4, 3, 0)
y = c(5 ,6 , 4 ,3)
Xy = data.frame(x = x, y = y)
```

```r
lm_xy = lm(Xy$x~Xy$y)
summary(lm_xy)$r.squared
```

```
## [1] 0.5
```

```r
summary(lm_xy)$sigma
```

```
## [1] 1.581139
```

Create a dataset $\mathbb{D}$ which we call `Xy` such that the linear model as R^2 about 0% but x, y are clearly associated.

```r
x2 = c(x)
y2 = c(rep(0,1,2),17)
Xy2 = data.frame(x = x2, y = y2)
```

```
## Error in data.frame(x = x2, y = y2): arguments imply differing number of rows: 4, 3
```

```r
lm_xy2 = lm(Xy2$y~Xy2$x)
```

```
## Error in eval(predvars, data, env): object 'Xy2' not found
```

```r
summary(lm_xy2) $r.squared
```

```
## Error in summary(lm_xy2): object 'lm_xy2' not found
```

```r
summary(lm_xy2) $sigma
```

```
## Error in summary(lm_xy2): object 'lm_xy2' not found
```

Load up the famous iris dataset and drop the data for Species "virginica".

```r
data(iris)
?iris
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

```r
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
iris_new1 = as.data.frame(iris[iris$Species != "Virginica",])
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

```r
summary(iris_new1)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
## 
```

If the only input x is Species and you are trying to predict y which is Petal.Length, what would a reasonable, naive prediction be under both Species? Hint: it's what we did in class.

```r
x = iris_new1$Species
y = iris_new1$Petal.Length
sum_reff_cat = 0
sum_alt_cat = 0
n = numeric()
for(i in 1 : length(x)){
  if(x[i] == 'sertosa'){
    sum_reff_cat = sum_reff_cat + y[i]
    n = i
  } else{
    sum_alt_cat = sum_alt_cat + y[i]

  }
}
b_0 = sum_reff_cat/n
b_1 = sum_alt_cat/(length(x)-n) - b_0
```

Prove that this is the OLS model by fitting an appropriate `lm` and then using the predict function to verify you get the same answers as you wrote previously.

```
lm_petal_length = lm(iris_new1$Petal.Length ~ iris_new1$Species)
new_input = data.frame(x = rep(c("setosa" , "versicolor"), each = 50))
predict(lm_petal_length, newdata = data.frame(x =rep(c("setosa" , "versicolor"), each = 50)))
```

```
## Warning: 'newdata' had 100 rows but variables found have 150 rows
```

```
##     1     2     3     4     5     6     7     8     9    10    11    12
## 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462
##    13    14    15    16    17    18    19    20    21    22    23    24
## 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462
##    25    26    27    28    29    30    31    32    33    34    35    36
## 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462
##    37    38    39    40    41    42    43    44    45    46    47    48
## 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462 1.462
##    49    50    51    52    53    54    55    56    57    58    59    60
## 1.462 1.462 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260
##    61    62    63    64    65    66    67    68    69    70    71    72
## 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260
##    73    74    75    76    77    78    79    80    81    82    83    84
## 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260
##    85    86    87    88    89    90    91    92    93    94    95    96
## 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260 4.260
##    97    98    99   100   101   102   103   104   105   106   107   108
## 4.260 4.260 4.260 4.260 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552
##   109   110   111   112   113   114   115   116   117   118   119   120
## 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552
##   121   122   123   124   125   126   127   128   129   130   131   132
## 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552
##   133   134   135   136   137   138   139   140   141   142   143   144
## 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552 5.552
##   145   146   147   148   149   150
## 5.552 5.552 5.552 5.552 5.552 5.552
```