Weekly Assignment 6

Sean Leggett - BDA201 Winter 2020 March 14, 2020

Questions

1. Calculate the linear regression line for the data and plot it on a scatter plot along with the data.

2. What is the slope? What is the y-intercept? Provide the units of each.

3. Give a practical interpretation of what the slope and the y-intercept mean in the context of the problem.
   Give a separate interpretation for each, and be sure to write in complete, grammatically correct sentences.

4. Using the linear regression model, estimate the number of components that need to be repaired, if a customer spends an hour on the phone for a service call (round to the nearest whole number). Is this interpolation or extrapolation?

5. Suppose that a certain customer has an issue that requires 12 components to be fixed. How long does your regression model suggest that the customer will have to spend on the phone during a service call addressing this issue? Is this interpolation or extrapolation?

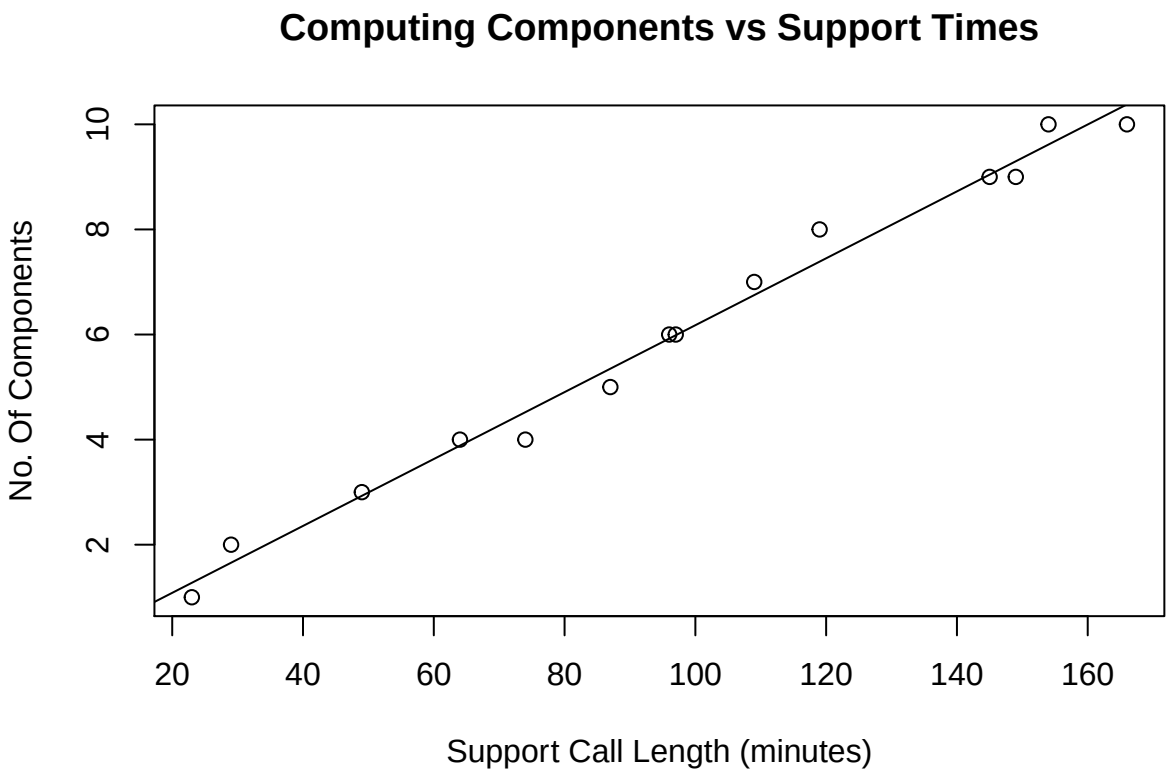
Now you also include another variable (age of computer in months) to the dataset. Make a new multiple linear regression model including the new variable and show whether addition of new variable to the model was a good idea? What is the adjusted R2 of the model?

Answers

```
## setup data
compsci <- data.frame("Call_Length" = c(23, 29, 49, 64, 74, 87, 96, 97, 109, 119, 149, 145, 154, 166),
                      "Components" = c(1, 2, 3, 4, 4, 5, 6, 6, 7, 8, 9, 9, 10, 10)
)
```

Calculate linear regression line:

```
## establish model
supportlm <- lm(Components ~ Call_Length, data = compsci)
## plot data
supportplot <- plot(compsci$Call_Length, compsci$Components,
                    main = "Computing Components vs Support Times",
                    xlab = "Support Call Length (minutes)",
                    ylab = "No. Of Components")
## add model to plot
abline(supportlm)
```




```
supportplot
```

```
## NULL
```

The assignment does not call for it but we are curious about correlation here as our data seem to really closely match regression line. The correlation turns out to be significant, as expected.

```
cor.test(compsci$Components, compsci$Call_Length)
```

```
##
##  Pearson's product-moment correlation
##
## data:  compsci$Components and compsci$Call_Length
## t = 30.712, df = 12, p-value = 8.916e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9795996 0.9980632
## sample estimates:
##       cor
## 0.9936987
```

Slope and Y-Intercept:
The code below provides values for slope and y-intercept. Simply calling the linear model returns our intercept of -0.18959 and slope of 0.06367

```r
summary(supportlm)
```

```
##
## Call:
## lm(formula = Components ~ Call_Length, data = compsci)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52196 -0.29158  0.04171  0.21589  0.61291
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.189594   0.221682  -0.855    0.409
## Call_Length  0.063670   0.002073  30.712 8.92e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3455 on 12 degrees of freedom
## Multiple R-squared:  0.9874, Adjusted R-squared:  0.9864
## F-statistic: 943.2 on 1 and 12 DF,  p-value: 8.916e-13
```

Meaning of Slope and Y-Intercept:
In a general sense, slope and y-intercept allow us to draw a straight line over a graph containing an x-axis and a y-axis, such as the one produced in our plot of support call length. The y-intercept is the starting point on the y-axis for our line. The slope is the amount of change in Y for each incremental step along the x-axis. In our case, for every step along the x-axis (one additional minute of call duration), we would multiply by 0.06367 to obtain the y-axis value (number of components) of a datapoint used to build the line.

In a more specific context, the formula for any straight line on a graph is y = mx +b, where ('m' = slope and 'b' = y-intercept). Therefore, we can calculate any value of y on a line for any value of x by multiplying x by slope and adding the y-intercept.

To illustrate, we can try the formula and compare to our plot. For example we will use 80 minutes call duration which would result in: y = 0.06367 x 80 + -0.18959. The result is 4.90401. We can confirm this looking at our plot when looking straight up from 80 on the x-axis and observing that it passes through ~4.9 on the y-axis.

Ultimately, knowing the slope and y-intercept allows us to forecast any value of x given y, or any value of y, given x.

Forecast components for an hour long call:
If a customer had an hour long call (60 minutes) we can use our knowledge of this model's slope and y-intercept to forecast the number of components they own. While the actual value for purposes of plotting a line is equal to 3.60582, we round to 4 components. We consider this to be interpolation as we are establishing a value at a point which lies within the range of our observed data.

```r
## setup new value
newdata <- data.frame("Call_Length" = 60)
## use predict function for the lm and pass newdata
compred <- predict(supportlm, newdata = newdata)
compred
```

```
##        1
## 3.630582
```

Predicting X:
If a customer had 12 components, how long would we anticipate their support calls to last? We were unable to find a satisfactory method for predicting X in our model without changing the relationship and building a separate LM. As such, we decided to use algebra to solve for the answer. The answer is 188 minutes. We consider this extrapolation since it forecasts a value outside of our observed data range.

y = mx + b 12 = 0.06367(x) + -0.18959 188.4718 = x + -0.18959 x = 188 minutes

Multivariate Regression:
For sake of clarity we found it easier to reproduce dataframe with added column as a new object.

```
expandcomp <- data.frame("Call_Length" = c(23, 29, 49, 64, 74, 87, 96, 97, 109, 119, 149, 145, 154, 166
                         "Components" = c(1, 2, 3, 4, 4, 5, 6, 6, 7, 8, 9, 9, 10, 10),
                         "Age" = c(20, 21, 19, 22, 24, 25, 28, 31, 31, 33, 34, 30, 38, 34)
)
```

New model...

```
explm <- lm(Components ~ Call_Length + Age, data = expandcomp)
summary(explm)
```

```
##
## Call:
## lm(formula = Components ~ Call_Length + Age, data = expandcomp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44161 -0.22809  0.08663  0.23638  0.42090
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.422191   0.632768  -2.248   0.0461 *
## Call_Length  0.054410   0.004878  11.153 2.46e-07 ***
## Age          0.076561   0.037351   2.050   0.0650 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3069 on 11 degrees of freedom
## Multiple R-squared:  0.9909, Adjusted R-squared:  0.9893
## F-statistic: 599.5 on 2 and 11 DF,  p-value: 5.919e-12
```

Our initial model showed extremely strong correlation. Accordingly, there is not a lot of room for improving the model significantly. However, adding computer age did indeed yield improved predictive quality of the model. Adjusted R-squared increased to 0.9893 from 0.9864. We also see a decrease in P-Value to almost infinitessimal notation. If we understood our research correctly, this is an even stronger P-Value than that accepted as part of the Higgs-Boson discovery.