Unit 9 Homework: Large-Sample Regression Theory

w203: Statistics for Data Science

What Makes a Successful Video Game?

The file `video_games.csv` contains data on 1212 video games that were on sold in 2011. It was compiled by Joe Cox, an economist at the University of Portsmouth.

Three key variables are as follows:

Variable	Meaning
Metrics.Sales	The total sales, measured in millions of dollars.
Metrics.Review.Score	Metacritic review score, an indicator of quality, out of 100.
Length.Completionists.Average	The mean time that players reported completing everything in the game, in hours.

You can find an explanation of other variables at https://think.cs.vt.edu/corgis/csv/video_games/.

You want to fit a regression predicting `Metrics.Sales`, with `Metrics.Review.Score` and `Length.Completionists.Average` as predictors.

0. Rename the variables that you are going to use to something sensible – variable names that have both periods and capital letters are not sensible. :fire: Better would be, for example changing `Metrics.Sales` to just `sales`.

```
dat<-read.csv('video_games.csv')
df<-dat %>%
  select('Metrics.Sales', 'Metrics.Review.Score', 'Length.Completionists.Average')%>%
  rename(sales = Metrics.Sales, score = Metrics.Review.Score,
         length = Length.Completionists.Average )
```

1. Examining the data, and using your background knowledge, evaluate the assumptions of the large-sample linear model.

Examining the data:

```
summary(df)
```

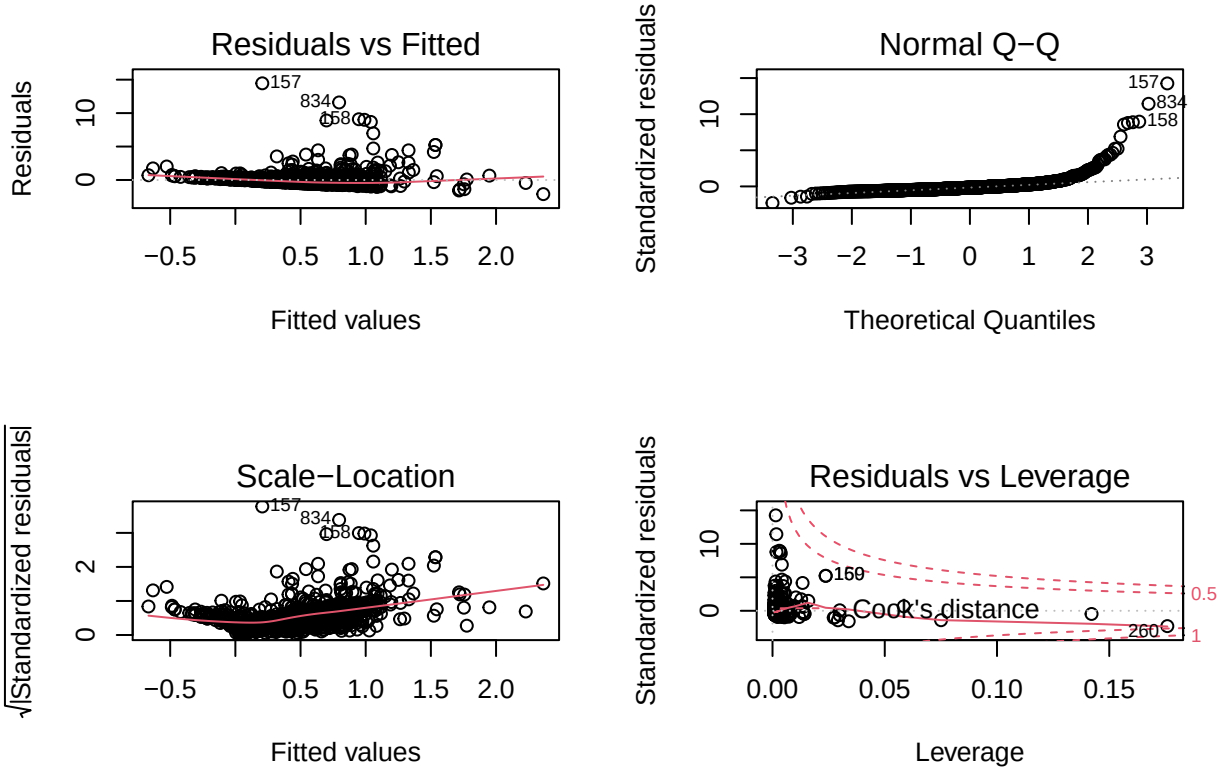
```
##      sales             score           length
##  Min.   : 0.0100   Min.   :19.00   Min.   :  0.00
##  1st Qu.: 0.0900   1st Qu.:60.00   1st Qu.:  0.00
##  Median : 0.2100   Median :70.00   Median :  6.00
##  Mean   : 0.5032   Mean   :68.83   Mean   : 19.81
```

```
##  3rd Qu.: 0.4600   3rd Qu.:79.00   3rd Qu.: 21.55
##  Max.   :14.6600   Max.   :98.00   Max.   :683.13
```

To evaluate the assumptions of the large-sample linear model, I have made 4 regression plots along with the interpretations to them.

In Linear model the sample size rule of thumb is that the regression analysis requires at least 20 cases per independent variable in the analysis. Linear model has five key assumptions:

- Linear relationship
- Multivariate normality
- No or little multicollinearity
- No auto-correlation
- Homoscedasticity




```
##
## Call:
## lm(formula = df$sales ~ df$score + df$length)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1125 -0.4223 -0.1852  0.0918 14.4534
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.0919732  0.1592648  -6.856 1.12e-11 ***
## df$score     0.0223899  0.0023092   9.696  < 2e-16 ***
## df$length    0.0027297  0.0006416   4.255 2.25e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.015 on 1209 degrees of freedom
## Multiple R-squared:  0.1022, Adjusted R-squared:  0.1007
## F-statistic: 68.79 on 2 and 1209 DF,  p-value: < 2.2e-16
```

1. Fig.1 is interpretation of residual vs fitted regression plot: This scatter plot shows the distribution of residuals (errors) vs fitted values (predicted values). It reveals various useful insights including outliers. If there exist any pattern (may be, a parabolic shape) in this plot, consider it as signs of non-linearity in the data. Here in Fig.1, a funnel shape is evident in the plot, I consider it as the signs of non constant variance i.e. heteroskedasticity.

2. Fig.2 is normal q-q plot regression interpretation: This q-q or quantile-quantile is a scatter plot which helps to validate the assumption of normal distribution in a data set. Using this plot we can infer if the data comes from a normal distribution. If yes, the plot would show fairly straight line. Absence of normality in the errors can be seen with deviation in the straight line. Here in Fig.2, the plot show a fairly straight line, confirm multivariate normality.

3. Fig.3 is scale location regression plot: This plot is also used to detect homoskedasticity (assumption of equal variance). It shows how the residual are spread along the range of predictors. It's similar to residual vs fitted value plot except it uses standardized residual values. Ideally, there should be no discernible pattern in the plot. This would imply that errors are normally distributed. But, here in the case of Fig.3, the plot shows discernible pattern (link a funnel shape), it would imply non-normal distribution of errors.

4. Fig.4 is residual vs leverage regression plot interpretation: Cook's distance attempts to identify the points which have more influence than other points. Such influential points tends to have a sizable impact of the regression line. In other words, adding or removing such points from the model can completely change the model statistics. Therefore, in this plot, the large values marked by cook's distance might require further investigation. For influential observations which are nothing but outliers, if not many, we can remove those rows.

2. Whether you consider the large-sample linear model sufficiently valid or not, proceed to fit the linear model using `lm()`.

```
library(gvlma)

fit <- lm(df$sales ~ df$score + df$length)
summary(fit)
```

```
##
## Call:
## lm(formula = df$sales ~ df$score + df$length)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1125 -0.4223 -0.1852  0.0918 14.4534
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.0919732  0.1592648  -6.856 1.12e-11 ***
## df$score     0.0223899  0.0023092   9.696  < 2e-16 ***
```

```
## df$length    0.0027297  0.0006416   4.255 2.25e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.015 on 1209 degrees of freedom
## Multiple R-squared:  0.1022, Adjusted R-squared:  0.1007
## F-statistic: 68.79 on 2 and 1209 DF,  p-value: < 2.2e-16
```

```
gvmodel <- gvlma(fit)
summary(gvmodel)
```

```
##
## Call:
## lm(formula = df$sales ~ df$score + df$length)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1125 -0.4223 -0.1852  0.0918 14.4534
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.0919732  0.1592648  -6.856 1.12e-11 ***
## df$score     0.0223899  0.0023092   9.696  < 2e-16 ***
## df$length    0.0027297  0.0006416   4.255 2.25e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.015 on 1209 degrees of freedom
## Multiple R-squared:  0.1022, Adjusted R-squared:  0.1007
## F-statistic: 68.79 on 2 and 1209 DF,  p-value: < 2.2e-16
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = fit)
##
##                       Value   p-value                   Decision
## Global Stat        262748.77 0.000e+00 Assumptions NOT satisfied!
## Skewness            10030.69 0.000e+00 Assumptions NOT satisfied!
## Kurtosis           252418.54 0.000e+00 Assumptions NOT satisfied!
## Link Function          49.76 1.738e-12 Assumptions NOT satisfied!
## Heteroscedasticity    249.78 0.000e+00 Assumptions NOT satisfied!
```

```
#
# library(car)
# crPlots(fit)
#
# qqPlot(fit, labels = row.names(df), id.method = 'identify', simulate = TRUE, main = 'Q-Q Plot')
#
# ncvTest(fit)
```

```
# durbinWatsonTest(fit)
# vif(fit)
#
# fit <- lm(df$sales ~ df$score)
# summary(fit)
#
# gvmodel <- gvlma(fit)
# summary(gvmodel)
```

- Meaning Behind Each Section of Summary(): Here's a brief description: (1) Call: This is an R feature that shows what function and parameters were used to create the model. (2) Residuals: Difference between what the model predicted and the actual value of y. You can calculate the Residuals section like so: summary(y-model$fitted.values); (3) Coefficients: These are the weights that minimize the sum of the square of the errors. (4) Std. Error is Residual Standard Error (see below) divided by the square root of the sum of the square of that particular x variable. (5) t value: Estimate divided by Std. Error (6) Pr(>|t|): Look up your t value in a T distribution table with the given degrees of freedom. With those sections out of the way, we'll focus on the bottom of the summary output.
- Residual Standard Error : In R, the lm summary produces the standard deviation of the error with a slight twist. Standard deviation is the square root of variance. Standard Error is very similar. The only difference is that instead of dividing by n-1, you subtract n minus 1 + # of variables involved.
- Multiple R-Squared: Also called the coefficient of determination, this is an oft-cited measurement of how well your model fits to the data. While there are many issues with using it alone (see Anscombe's quartet) , it's a quick and pre-computed check for your model. R-Squared subtracts the residual error from the variance in Y. The bigger the error, the worse the remaining variance will appear. If you notice, numerator doesn't have to be positive. If the model is so bad, you can actually end up with a negative R-Squared.
- Adjusted R-Squared: Multiple R-Squared works great for simple linear (one variable) regression. However, in most cases, the model has multiple variables. The more variables you add, the more variance you're going to explain. So you have to control for the extra variables. Adjusted R-Squared normalizes Multiple R-Squared by taking into account how many samples you have and how many variables you're using. Notice how k is in the denominator. If you have 100 observations (n) and 5 variables, you'll be dividing by 100-5-1 = 94. If you have 20 variables instead, you're dividing by 100-20-1 = 79. As the denominator gets smaller, the results get larger: 99 /94 = 1.05; 79/94 = 1.25. A larger normalizing value is going to make the Adjusted R-Squared worse since we're subtracting its product from one.
- F-Statistic: Finally, the F-Statistic. Including the t-tests, this is the second "test" that the summary function produces for lm models. The F-Statistic is a "global" test that checks if at least one of your coefficients are nonzero.

3. Examine the coefficient for Metrics.Review.Score and give an interpretation of what it means.

```
scoresales.lm <- lm(sales ~ score, data=df)

#scorelength.lm <- lm(score ~ length, data=df)

summary(scoresales.lm)
```

```
##
```

```
## Call:
## lm(formula = sales ~ score, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8663 -0.4398 -0.2052  0.0973 14.4232
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.189765   0.158708  -7.497 1.26e-13 ***
## score        0.024596   0.002266  10.854  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.022 on 1210 degrees of freedom
## Multiple R-squared:  0.08873,    Adjusted R-squared:  0.08797
## F-statistic: 117.8 on 1 and 1210 DF,  p-value: < 2.2e-16
```

```
#$r.squared
#summary(scorelength.lm)$r.squared
```

The most common interpretation of the coefficient of determination is how well the regression model fits the observed data. For example, a coefficient of determination of 60% shows that 60% of the data fit the regression model. Generally, a higher coefficient indicates a better fit for the model.

However, it is not always the case that a high r-squared is good for the regression model. The quality of the coefficient depends on several factors, including the units of measure of the variables, the nature of the variables employed in the model, and the applied data transformation. Thus, sometimes, a high coefficient can indicate issues with the regression model.

No universal rule governs how to incorporate the coefficient of determination in the assessment of a model. The context in which the forecast or the experiment is based is extremely important, and in different scenarios, the insights from the statistical metric can vary.

```
#bptest(scoresales.lm)
#bptest(scorelength.lm)
```

4. Perform a hypothesis test to assess whether video game quality has a relationship with total sales. Please use `vcovHC` from the `sandwich` package with the default options ("HC3") to compute robust standard errors. To conduct the test, use `coeftest` from the `lmtest` package.

```
library(lmtest)
m1 <- lm(df$sales ~ df$score)
#bptest(m1)
#ptest(m1, studentsize=FALSE)

library(sandwich)
#summary(m1)
#NeweyWest(m1)

#result1<-coeftest(m1, vcov = NeweyWest(m1))
result2<-coeftest(m1, vcov. = vcovHC, type = "HC3")
```

```
print(result2)
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -1.1897654  0.1844771 -6.4494  1.62e-10 ***
## df$score     0.0245963  0.0029085  8.4567 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#summary(m1)
```

5. How many more sales does your model predict for a game one standard-deviation higher than the mean review, vs. a game one standard-deviation lower than the mean review, holding all else equal? Answer this in two different ways:

(a) Compute the standard deviation of the review score, and multiply the appropriate model coefficient by two-times this standard deviation.

```
model <- lm(sales ~ score + length , data = df)

#m2 <- lm(sales ~ score , data = df)

model
```

```
##
## Call:
## lm(formula = sales ~ score + length, data = df)
##
## Coefficients:
## (Intercept)        score       length
##    -1.09197      0.02239      0.00273
```

```
#m2
a <- sd(df$score)
b <- sd(df$length)
a
```

```
## [1] 12.95627
```

```
b
```

```
## [1] 46.63455
```

```
c <- a * 2 * 0.0224
c
```

```
## [1] 0.5804407
```

(b) Use the `predict` function with the model that you have estimated. You can read the documentation for `predict.lm` which is the predict method for linear model objects (the type that you have fit here). Include a data frame (that has the same variable names as the data frame that you fitted the model against) in the `newdata` argument to `predict`. This data frame should have two rows and two columns. The column for the reviews should change from $\mu - \sigma$ to $\mu + \sigma$; the column for the play time should be set to a constant, sensible level (perhaps the μ of this variable).

```
x<-df$score
mean(x)
```

```
## [1] 68.82838
```

```
sd(x)
```

```
## [1] 12.95627
```

```
x1 = mean(x) + sd(x)
x2 = mean(x) - sd(x)
x1
```

```
## [1] 81.78465
```

```
x2
```

```
## [1] 55.87212
```

```
y <-mean(df$length)
y
```

```
## [1] 19.80822
```

```
model <- lm(sales ~ score + length, data = df)

print(model)
```

```
##
## Call:
## lm(formula = sales ~ score + length, data = df)
##
## Coefficients:
## (Intercept)        score       length
##    -1.09197      0.02239      0.00273
```

```
summary(model)
```

```
##
## Call:
## lm(formula = sales ~ score + length, data = df)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1125 -0.4223 -0.1852  0.0918 14.4534
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.0919732  0.1592648  -6.856 1.12e-11 ***
## score        0.0223899  0.0023092   9.696  < 2e-16 ***
## length       0.0027297  0.0006416   4.255 2.25e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.015 on 1209 degrees of freedom
## Multiple R-squared:  0.1022, Adjusted R-squared:  0.1007
## F-statistic: 68.79 on 2 and 1209 DF,  p-value: < 2.2e-16
```

```
a <-predict.lm(model, newdata=data.frame(score = x1, length = y), interval = 'prediction', level = 0.95)
b <-predict.lm(model, newdata=data.frame(score = x2, length = y), interval = 'prediction', level = 0.95)
c <- a-b
c
```

```
##         fit       lwr       upr
## 1 0.5801799 0.5801799 0.5801799
```

5. **Optional:** Open the attached paper by Joe Cox, and read section 3. Which assumption did the author focus on, and why do you think that is?

Note: Maximum score on any homework is 100%