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.

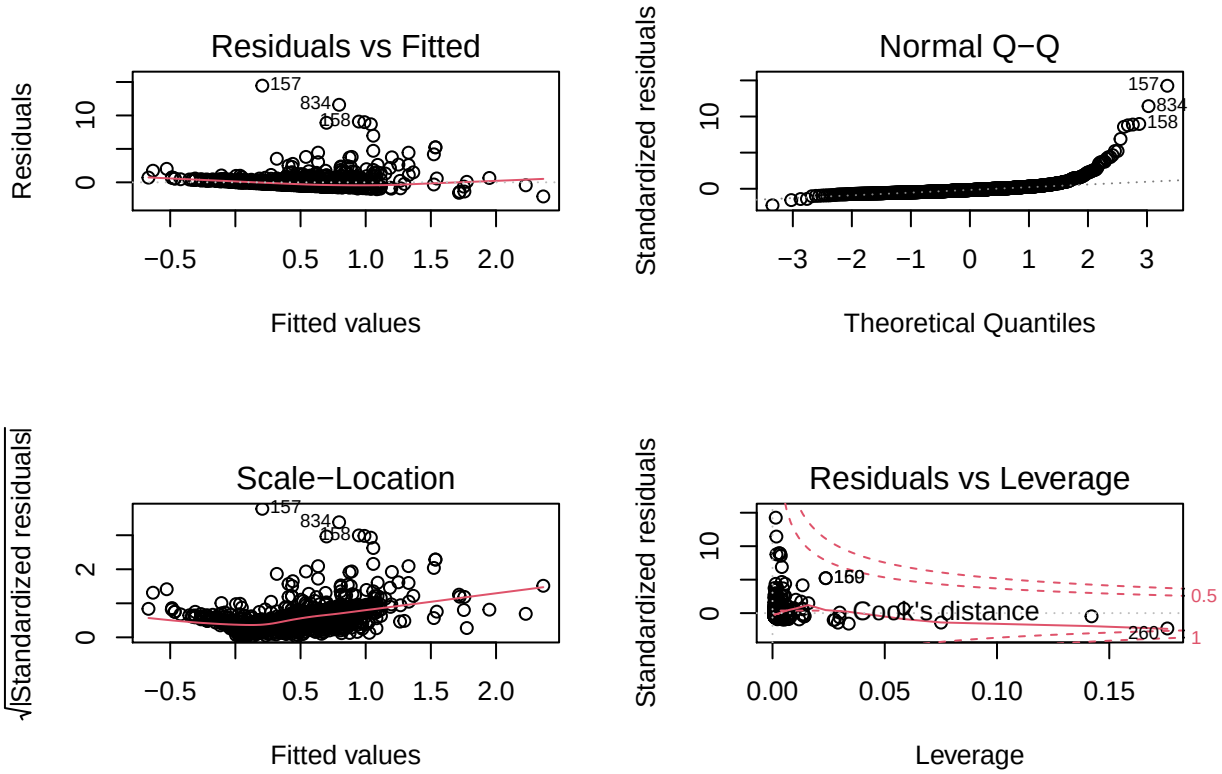
You can find an explanation of other variables at https://think.cs.vt.edu/corgis/csv/video_games/.

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`.

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

```
##      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
```



Linear regression is an analysis that assesses whether one or more predictor variables explain the dependent (criterion) variable. The regression has five key assumptions:

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

A note about sample size. In Linear regression the sample size rule of thumb is that the regression analysis requires at least 20 cases per independent variable in the analysis.

In the software below, its really easy to conduct a regression and most of the assumptions are preloaded and interpreted for you.

(1) Linear regression needs the relationship between the independent and dependent variables to be linear. It is also important to check for outliers since linear regression is sensitive to outlier effects. The linearity assumption can best be tested with scatter plots, the following two examples depict two cases, where no and little linearity is present.

(2) The linear regression analysis requires all variables to be multivariate normal. This assumption can best be checked with a histogram or a Q-Q-Plot. Normality can be checked with a goodness of fit test, e.g., the Kolmogorov-Smirnov test. When the data is not normally distributed a non-linear transformation (e.g., log-transformation) might fix this issue.

(3) Linear regression assumes that there is little or no multicollinearity in the data. Multicollinearity occurs when the independent variables are too highly correlated with each other. Multicollinearity may be tested with three central criteria: 1) Correlation matrix – when computing the matrix of Pearson's Bivariate Correlation among all independent variables the correlation coefficients need to be smaller than 1. 2) Tolerance – the tolerance measures the influence of one independent variable on

all other independent variables; the tolerance is calculated with an initial linear regression analysis. Tolerance is defined as $T = 1 - R^2$ for these first step regression analysis. With $T < 0.1$ there might be multicollinearity in the data and with $T < 0.01$ there certainly is. 3) Variance Inflation Factor (VIF) – the variance inflation factor of the linear regression is defined as $VIF = 1/T$. With $VIF > 5$ there is an indication that multicollinearity may be present; with $VIF > 10$ there is certainly multicollinearity among the variables. If multicollinearity is found in the data, centering the data (that is deducting the mean of the variable from each score) might help to solve the problem. However, the simplest way to address the problem is to remove independent variables with high VIF values.

(4) Linear regression analysis requires that there is little or no autocorrelation in the data. Autocorrelation occurs when the residuals are not independent from each other. In other words when the value of y(x+1) is not independent from the value of y(x). While a scatterplot allows you to check for autocorrelations, you can test the linear regression model for autocorrelation with the Durbin-Watson test. Durbin-Watson's d tests the null hypothesis that the residuals are not linearly auto-correlated. While d can assume values between 0 and 4, values around 2 indicate no autocorrelation. As a rule of thumb values of $1.5 < d < 2.5$ show that there is no auto-correlation in the data. However, the Durbin-Watson test only analyses linear autocorrelation and only between direct neighbors, which are first order effects.

(5) The last assumption of the linear regression analysis is homoscedasticity. The scatter plot is good way to check whether the data are homoscedastic (meaning the residuals are equal across the regression line). The following scatter plots show examples of data that are not homoscedastic (i.e., heteroscedastic): The Goldfeld-Quandt Test can also be used to test for heteroscedasticity. The test splits the data into two groups and tests to see if the variances of the residuals are similar across the groups. If homoscedasticity is present, a non-linear correction might fix the problem.

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)
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)
```

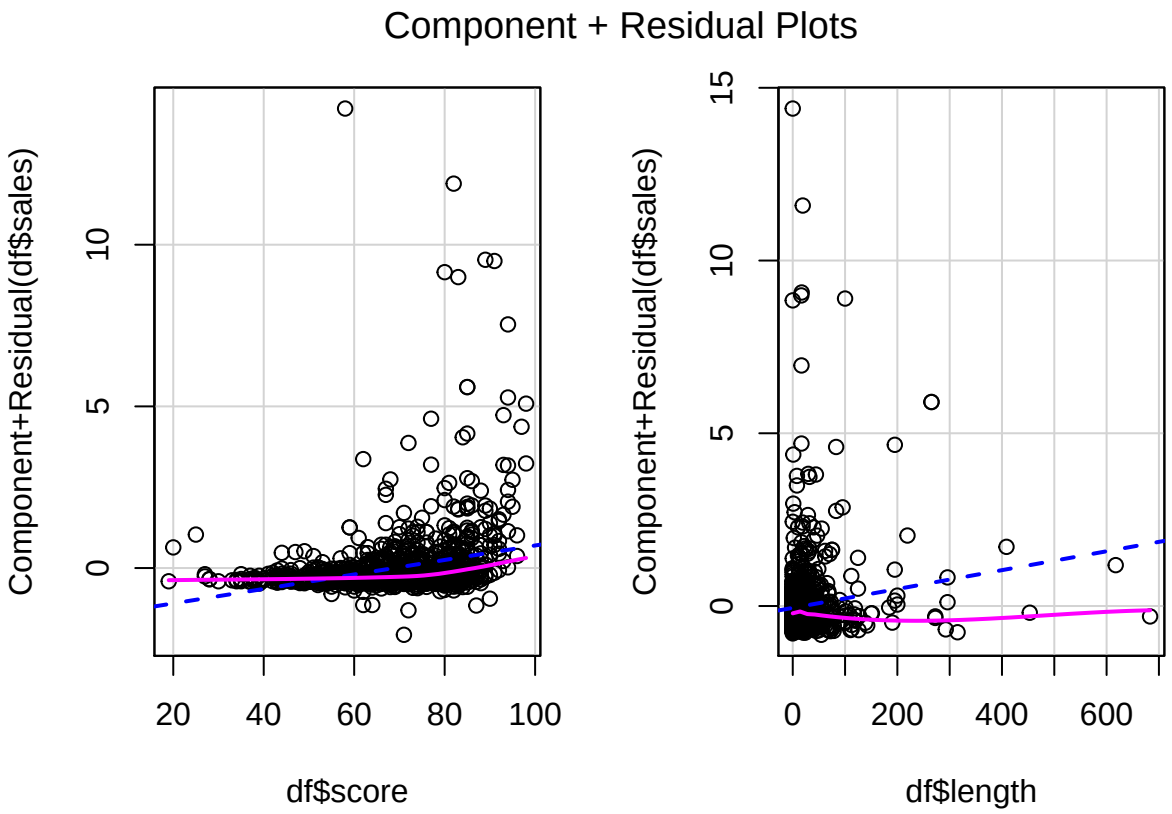
```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

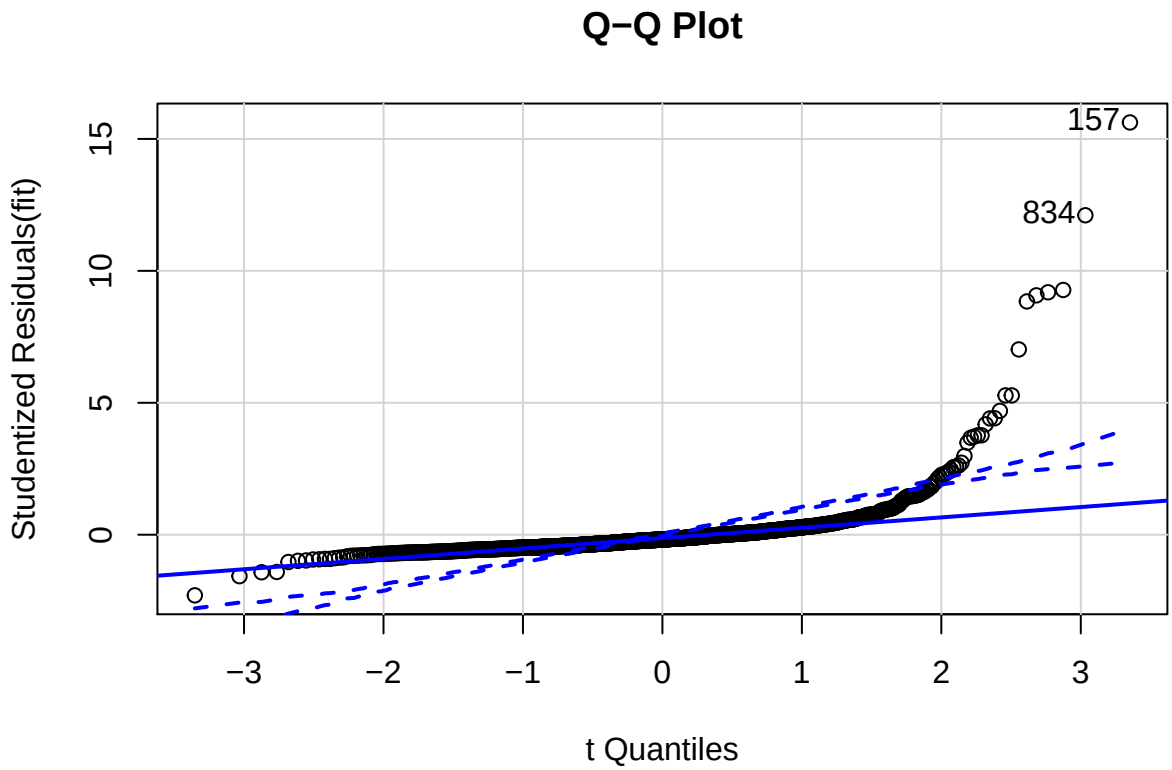
```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
crPlots(fit)
```




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




```
## [1] 157 834
```

```
ncvTest(fit)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 386.6485, Df = 1, p = < 2.22e-16
```

```
durbinWatsonTest(fit)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.6379745     0.7124182       0
##  Alternative hypothesis: rho != 0
```

```
vif(fit)
```

```
##  df$score df$length
##  1.053108  1.053108
```

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

```
scoresales.lm <- lm(score ~ sales, data=df)
scorelength.lm <- lm(score ~ length, data=df)
summary(scoresales.lm)$r.squared
```

```
## [1] 0.0887267
```

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

```
## [1] 0.05042935
```

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)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  scoresales.lm
## BP = 41.037, df = 1, p-value = 1.494e-10
```

```
bptest(scorelength.lm)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  scorelength.lm
## BP = 14.348, df = 1, p-value = 0.0001519
```

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$score ~ df$sales)
#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) 67.01333    0.51107 131.1234 < 2.2e-16 ***
## df$sales     3.60731    0.85261   4.2309 2.503e-05 ***
## ---
## 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)
model
```

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

```
##
## Coefficients:
## (Intercept)        score       length
##    -1.09197      0.02239      0.00273
```

```
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
```

```r
# d3 <- filter(df, df$score > 68.83 |  df$score < 55.87  )
# d1 <- filter(df, df$score > 68.83)
# d2 <- filter(df, df$score < 55.87)
#
#
# nrow(d1)
# nrow(d2)
# nrow(d3)
# '''

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
```

```r
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
```

```r
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%