Simple Regression

This R markdown document provides an example of performing a simple regression using the lm() function in R and compares the output with the linReg() function in the jmv (Jamovi) package.

Package management in R

``` r
# keep a list of the packages used in this script
packages <- c("tidyverse","rio","jmv")
```

This next code block has eval=FALSE because you don’t want to run it when knitting the file. Installing packages when knitting an R notebook can be problematic.

``` r
# check each of the packages in the list and install them if they're not installed already
for (i in packages){
  if(! i %in% installed.packages()){
    install.packages(i,dependencies = TRUE)
  }
  # show each package that is checked
  print(i)
}
```

``` r
# load each package into memory so it can be used in the script
for (i in packages){
  library(i,character.only=TRUE)
  # show each package that is loaded
  print(i)
}
```

```
    ## -- Attaching packages --------------------------------------- tidyverse
1.3.0 --

    ## v ggplot2 3.3.3     v purrr   0.3.4
    ## v tibble  3.0.6     v dplyr   1.0.4
    ## v tidyr   1.1.2     v stringr 1.4.0
    ## v readr   1.4.0     v forcats 0.5.1

    ## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
    ## x dplyr::filter() masks stats::filter()
    ## x dplyr::lag()    masks stats::lag()

    ## [1] "tidyverse"
    ## [1] "rio"
    ## [1] "jmv"
```

Simple Regression

Simple regression is predicting a continuous outcome variable (dependent variable) with a single continuous predictor variable (independent variable). You can perform regressions using categorical variable, but we’ll talk more about that later.

Open data file

The rio package works for importing several different types of data files. We’re going to use it in this class. There are other packages which can be used to open datasets in R. You can see several options by clicking on the Import Dataset menu under the Environment tab in RStudio. (For a csv file like we have this week we’d use either From Text(base) or From Text (readr). Try it out to see the menu dialog.)

``` r
# import the Week3.rds dataset into RStudio
# Using the file.choose() command allows you to select a file to import from another folder.
# dataset <- rio::import(file.choose())
# This command will allow us to import the rds file included in our project folder.
dataset <- rio::import("Album Sales.sav")
```

Get R code from Jamovi output

You can get the R code for most of the analyses you do in Jamovi.

1. Click on the three vertical dots at the top right of the Jamovi window.
2. Click on the Syndax mode check box at the bottom of the Results section.
3. Close the Settings window by clicking on the Hide Settings arrow at the top right of the settings menu.
4. you should now see the R code for each of the analyses you just ran.

lm() function in R

Many linear models are calculated in R using the lm() function. We’ll look at how to perform a simple regression using the lm() function since it’s so common.

Visualization

``` r
ggplot(dataset, aes(x = Adverts, y = Sales)) +
  geom_point() +
  stat_smooth(method = lm)
```

~~~
    ## `geom_smooth()` using formula 'y ~ x'

![](Simple-Regression-Assignment_files/figure-markdown_github/unnamed-
chunk-5-1.png)

#### Computation

``` r
model <- lm(formula = Sales ~ Adverts, data = dataset)
model
```

    ## 
    ## Call:
    ## lm(formula = Sales ~ Adverts, data = dataset)
    ## 
    ## Coefficients:
    ## (Intercept)      Adverts  
    ##   134.13994      0.09612

#### Model assessment

``` r
summary(model)
```

    ## 
    ## Call:
    ## lm(formula = Sales ~ Adverts, data = dataset)
    ## 
    ## Residuals:
    ##      Min       1Q   Median       3Q      Max 
    ## -152.949  -43.796   -0.393   37.040  211.866 
    ## 
    ## Coefficients:
    ##              Estimate Std. Error t value Pr(>|t|)    
    ## (Intercept) 1.341e+02  7.537e+00  17.799   <2e-16 ***
    ## Adverts     9.612e-02  9.632e-03   9.979   <2e-16 ***
    ## ---
    ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
    ## 
    ## Residual standard error: 65.99 on 198 degrees of freedom
    ## Multiple R-squared:  0.3346, Adjusted R-squared:  0.3313 
    ## F-statistic: 99.59 on 1 and 198 DF,  p-value: < 2.2e-16

#### Standardized residuals from lm()

You might notice lm() does not provide the standardized residuals. Those
must me calculated separately.

``` r
standardized = lm(scale(Sales) ~ scale(Adverts), data=dataset)
summary(standardized)
```
~~~

```
    ## 
    ## Call:
    ## lm(formula = scale(Sales) ~ scale(Adverts), data = dataset)
    ## 
    ## Residuals:
    ##      Min       1Q   Median       3Q      Max 
    ## -1.89531 -0.54271 -0.00487  0.45900  2.62538 
    ## 
    ## Coefficients:
    ##                  Estimate Std. Error t value Pr(>|t|)    
    ## (Intercept)    -2.141e-17  5.782e-02   0.000        1    
    ## scale(Adverts)  5.785e-01  5.797e-02   9.979   <2e-16 ***
    ## ---
    ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
    ## 
    ## Residual standard error: 0.8177 on 198 degrees of freedom
    ## Multiple R-squared:  0.3346, Adjusted R-squared:  0.3313 
    ## F-statistic: 99.59 on 1 and 198 DF,  p-value: < 2.2e-16
```

function in Jamovi

Compare the output from the lm() function with the output from the function in the jmv package.

~~~
``` r
jmv::linReg(
  data = dataset,
  dep = Sales,
  covs = Adverts,
  blocks = list(list("Adverts")),
  refLevels = list(),
  modelTest = TRUE,
  anova = TRUE,
  ci = TRUE,
  stdEst = TRUE,
  ciStdEst = TRUE)
```
~~~

```
    ## 
    ##  LINEAR REGRESSION
    ## 
    ##  Model Fit Measures
    ## 
---------------------------------------------------------------------------
    ##    Model    R            R²           F            df1    df2    p 
    ## 
---------------------------------------------------------------------------
    ##        1    0.5784877    0.3346481    99.58687      1    198    < 
.0000001 
    ## 
---------------------------------------------------------------------------
    ## 
    ## 
    ##  MODEL SPECIFIC RESULTS
    ## 
```

```
    ##  MODEL 1
    ##
    ##  Omnibus ANOVA Test
    ##
-------------------------------------------------------------------------------
    ##                 Sum of Squares    df     Mean Square    F           p
    ##
-------------------------------------------------------------------------------
    ##    Adverts            433687.8      1     433687.833    99.58687    <
.0000001
    ##    Residuals          862264.2    198       4354.870
    ##
-------------------------------------------------------------------------------
    ##    Note. Type 3 sum of squares
    ##
    ##
    ##  Model Coefficients - Sales
    ##
------------------------------------------------------------------------------------------
    ##    Predictor    Estimate        SE             Lower           Upper
t            p             Stand. Estimate    Lower        Upper
    ##
------------------------------------------------------------------------------------------
    ##    Intercept    134.13993781    7.536574679    119.27768082
149.0021948    17.798528    < .0000001
    ##    Adverts        0.09612449    0.009632366      0.07712929
0.1151197     9.979322    < .0000001          0.5784877    0.4641726
0.6928029
    ##
------------------------------------------------------------------------------------------
```