Chapter 7. Regression

Daehyeog Lee

2023-10-02

Table of contents

7.0. Importing libraries

```
library(ggplot2); library(boot); library(car); library(QuantPsyc)
```

```
Loading required package: carData
```

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

```
The following object is masked from 'package:boot':

    logit
```

```
Loading required package: dplyr
```

```
Attaching package: 'dplyr'
```

```
The following object is masked from 'package:car':

    recode
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

Loading required package: purrr

Attaching package: 'purrr'

The following object is masked from 'package:car':

    some

Loading required package: MASS

Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select

Attaching package: 'QuantPsyc'

The following object is masked from 'package:base':

    norm
```

7.1. What will this chapter tell me?

A regression analysis on the data would enable us to predict future evaluations based on values of the predictor variables.

7.2. An introduction to regression

The essence of regression analysis is that we fit a model to our data and use it to predict values of the dependent variable from one or more independent variables. Predicting an outcome variable from one predictor variable is called **simple regression**, and predicting it from several predictor variables is called **multiple regression.** In regression, the model we fit is linear, which means that we summarize a data set with a straight line. We use a mathematical technique called the *method of least squares* to establish the line that best describes the data collected.

7.2.1. Some important information about straight lines

$Y_i = (b_0 + b_i X_i) + \epsilon_i$

Y_i is the outcome that we want to predict and X_i is the ith participant's score on the predictor variable. b_0 and b_1 are known as the regression coefficients, where b_1 is the gradient of the straight line and b_0 is the intercept of that line. The gradient of the line tells us something about the nature of the relationship being described and the intercept tells us where the model is. A residual term ϵ_i represents the difference between the score predicted by the line for participant i and the score that participant i actually obtained.

7.2.2. The method of least squares

The vertical differences between the line and the actual data are called residuals, which are the same as deviations. The lowest sum of squared residuals is the line of best fit. This 'line of best fit' is called a **regression model**.

7.2.3. Assessing the goodness of fit: sums of squares, *R* and R^2

The **total sum of squares** (SS_T) represents how good the mean is as a model of the observed data. The **residual sum of squares** (SS_R) represents the degree of inaccuracy when the best model is fitted to the data. The **model sum of squares** (SS_M) is obtained by calculating the difference between SS_T and SS_R. If the value of SS_M is large then the regression model is very different from using the mean to predict the outcome variable. This implies that the regression model has made a big improvement to how well the outcome variable can be predicted. However, if SS_M is small then using the regression model is little better than using the mean.

$R^2 = \frac{SS_M}{SS_T}$

R^2 represents the amount of variance in the outcome explained by the model (SS_M) relative to how much variation there was to explain in the first place (SS_T). Therefore, it represents the percentage of the variation in the outcome that can be explained by the model. This R^2 is also a squared value of Pearson's correlation coefficient.

A second use of the sums of squares in assessing the model is through the F-test. Test statistics like F are usually the amount of systematic variance divided by the amount of unsystematic variance.

$F = \frac{MS_M}{MS_R}$

F ratio is a measure of how much the model has improved the prediction of the outcome compared to the level of inaccuracy of the model. If a model is good, then we expect the improvement in prediction due to the model to be large (large MS_M), amd the difference

between the model and the observed data to be small (small MS_R). Because the sums of squares depend on the number of differences that we have added up, we use the average sums of squares (MS). For SS_M the degrees of freedom are simply the number of variables in the model, and for SS_R they are the number of observations minus the number of parameters being estimated.

7.2.4. Assessing individual predictors

The predictor in a regression model has a coefficient b_1, which in simple regression represents the gradient of the regression line. A regression coefficient of 0 means a unit change in the predictor variable results in no change in the predicted value of the outcome, and the gradient of the regression line is 0. The ***t*-statistic** tests the null hypothesis that the value of b is 0.

What we're interested in here is whether the b we have is big compared to the amount of error in that estimate. The standard error tells us something about how different b-values would be across different samples.

$t = \frac{b_o bserved}{SE_b}$

The degrees of freedom are $N - p - 1$, where N is the total sample size and p is the number of predictors. **R** provides the exact probability that the observed value of t would occur if the value of b was, in fact, 0. If this observed significance is less than .05, then we can assume that b is significantly different from 0.

7.3. Packages used in this chapter

"boot", "car" and "QuantPsyc" packages are used in this chapter.

7.4. General procedure for regression in R

7.4.1. Doing simple regression using R Commander

Using R Commander, we can load he data file by **Data -> Import data -> from text file, clipboard, or URL...** . Then we can choose **Statistics -> Fit models -> Linear regression** to activate the linear regression dialog box. On the left we choose a response variable (the outcome). On the right we choose an explanatory variable (the predictor).

7.4.2. Regression in R

We can run a regression analysis using the `lm()` function. The `lm()` function takes the general form:

```
newModel <- lm(outcome ~ predictor(s), data = dataFrame, na.action = an
action)
```

```
album1 <- read.delim("Album Sales 1.dat", header = TRUE)
albumSales.1 <- lm(album1$sales ~ album1$adverts)
# OR
albumSales.1 <- lm(sales ~ adverts, data = album1)
```

7.5. Interpreting a simple regression

We can show the object by executing:

```
summary(albumSales.1)
```

```
Call:
lm(formula = sales ~ adverts, data = album1)

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
```

7.5.1. Overall fit of the object model

From the output, `Multiple R-squared: 0.3346, Adjusted R-squared: 0.3313` provides the value of R^2 for the model. Because there is only one predictor, this value represents

the square of the simple correlation between advertising and album sales. We can find the square root of R^2 , 0.58, which is a Pearson correlation coefficient. The value of $R^2 = .335$ tells that advertising expenditure can account for 33.5% of the variation in album sales. This means that 67% of the variation in album sales cannot be explained by advertising alone.

Another part of the output, `F-statistic: 99.59 on 1 and 198 DF, p-value: < 2.2e-16`, gives the F-ratio. For these data, F = 99.59, which is significant at $p < .001$. Therefore, we can conclude that our regression model results in significantly better prediction of album sales than if we used the mean value of album sales.

7.5.2. Model parameters

The output labelled `Coefficients` contains the model parameters (the beta values). We can see that the intercept b_0 is 134.1, which can be interpreted as meaning that when no money is spent on advertising, the model predicts that 134,100 albums will be sold. The value of b_1 (`adverts`) represents the gradient of the regression line. Therefore, if our predictor variable variable is increased by one unit (if the advertising budget is increased by 1), then our model predicts that 0.096 units of extra albums will be sold. We can also see that the t-test tells us whether the b-value is different from 0. Since the probability of t-values occurring if the values of b in the population were 0 is less than .001 (<2e-16) we can tell that the bs are different from 0 and conclude that the advertising budget makes a significant contribution (p<.001) to predicting album sales.

7.5.3. Using the model

This significant model predicts that:

album sales = b_0 + b_1 advertising budget(i)

= 134.14 + (0.096 x advertising budget(i))

7.6. Multiple regression: the basics

Multiple regression is a logical extension in which there are several predictors.

$Y_i = (b_0 + b_1X_1i + b_2X_2i + ... + b_nX_ni\ ) + \epsilon_i$

Each predictor variable has its own coefficient, and the outcome variable is predicted from a combination of all the variables multiplied by their respective coefficients plus a residual term.

7.6.1. An example of a multiple regression model

We can make predictions based on multiple variables. We can draw a 3D scatterplot if we have two predictors. However, multiple regression can be used with three, four or even ten or more predictors.

7.6.2. Sums of squares, R and R^2

When there are several predictors we can't look at the simple R^2, but instead **R** produces a multiple R^2. Multiple R^2 is the square of the correlation between the observed values of Y and the values of Y predicted by the multiple regression model. It is the amount of variation in the outcome variable that is accounted for by the model.

7.6.3. Parsimony-adjusted measures of fit

One problem with R^2 is that it always go up when we add more variables to the model. The **Akaike information criterion (AIC)** penalizes the model for having more variables. However, there are no guidelines for how much larger is 'a lot' and how much larger is 'not very much' in AIC. Also, we can compare the AIC only between models of the same data.

7.6.4. Methods of regression

We have to carefully select the predictors for a model because the values of the regression coefficients depend upon the variables in the model.

7.6.4.1. Hierarchical

In **hierarchical regression** predictors are selected based on past work and the experimenter decides in which order to enter the predictors into the model.

7.6.4.2. Forced entry

Forced entry is a method in which all predictors are forced into the model simultaneously. Unlike hierarchical regression, the experimenter makes no decision about the order in which variables are entered.

7.6.4.3. Stepwise methods

In **stepwise regression** decisions about the order in which predictors are entered into the model are based on purely mathematical criterion. In *forward* method, an initial model is defined that contains only the intercept. The computer then searches for the predictor that best predicts the outcome variable. In *backward* method the computer begins by placing all predictors in the model and then by looking to see if the AIC goes down when each variable is removed. The backward method is preferable to the forward method because forward method has a higher risk of making a Type II error.

7.6.4.4. All-subsets methods

The problem with stepwise method is that they access the fit of a variable based on the other variables that were in the model. **All-subsets regression** tries every combination of variables, to see which one gives the best fit.

7.6.4.5. Choosing a method

Because of the danger of overfitting and underfitting, stepwise methods are best avoided except for exploratory model building. If we must do a stepwise regression then we have to **cross-validate** our model by splitting the data.

7.7. How accurate is my regression model?

If we find that our model is not generalizable, then we must restrict any conclusions based on the model to the sample used.

7.7.1. Assessing the regression model I: diagnostics

7.7.1.1. Outliers and residuals

An outlier is a case that differs substantially from the main trend of the data.It is important to detect outliers to see whether the model is biased in this way.

Residuals are known as the differences between the values of the outcome predicted by the model and the values of the outcome observed in the sample. The **normal** or **unstandardized residuals** are measured in the same units as the outcome variable, so they are difficult to interpret across different models. Instead, we us **standardized residuals**, which are the residuals divided by an estimate of their standard deviation. If more than 5% of cases have standardized residuals with an absolute value greater than 1.96, then there is an evidence that the model is a poor representation of the actual data.

7.7.1.2. Influential cases

One statistic that can be used to assess the influence of a particular case is the **adjusted predicted value**. The computer calculates a new model without a particular case and then uses this new model to predict the value of the outcome variable for the case that was excluded. When the residual is divided by the standard error it gives a standardized value known as the **studentized residual**. It is very useful to assess the influence of a case on the ability of the model to predict that case, but they do not provide any information about how a case influences the model as a whole. **Cook's distance** considers this, which is a measure of the overall influence of a case on the model.

Another measure of influence is **hat values (leverage)**, which gauge the influence of the observed value of the outcome variable over the predicted values. It is defined as $(k + 1/n)$, in which k is the number of predictors in the model and n is the number of participants. It lies between 0 and 1, in which 0 indicates that the case has no influence.

The difference between a parameter estimated using all cases and estimated when one case is excluded is known as the **DFBeta**.

7.7.1.3. A final comment on diagnostic statistics

Diagnostic statistics are not a way of justifying the removal of data points to effect some desirable change in the regression parameters.

7.7.2. Assessing the regression model II: generalization

For a regression model to generalize we must be sure that underlying assumptions have been met, and to test whether the model does generalize we can look at cross-validating it.

7.7.2.1. Checking assumptions

Several assumptions must be true to draw conclusions about a population based on a regression analysis done on a sample:

- **Variable types**: All predictor variables must be quantitative or categorical, and the outcome variable must be quantitative, continuous and unbounded.
- **Non-zero variance**: The predictors should have some variation in value.
- **No perfect multicollinearity**: There should be no perfect linear relationship between two or more of the predictors.
- **Predictors are uncorrelated with 'external variables'**: There should be no external variables that correlate with any of the variables included in the regression model.

- **Homoscedasticity**: At each level of the predictor variable, the variance of the residual terms should be constant.
- **Independent errors**: For any two observations the residual terms should be uncorrelated. This assumption can be tested with the **Durbin-Watson test**, which tests for serial correlations between errors.
- **Normally distributed errors**: It is assumed that the residuals in the model are random, normally distributed variables with a mean of 0. However, predictors do not need to be normally distributed.
- **Independence**: All of the values of the outcome variable ore independent.
- **Linearity**: It is assumed that the relationship we are modelling is a linear one.

7.7.2.2. Cross-validation of the model

Assessing the accuracy of a model across different samples is known as **cross-validation**.

- **Adjusted** R^2 : Whereas R^2 tells us how much of the variance in Y is accounted for by the regression model from our sample, the adjusted value tells us how much variance in y would be accounted for if the model had been derived from the population from which the sample was taken.
- **Data splitting**: This involves randomly splitting your dataset, computing a regression equation on both halves of the data and then comparing the resulting models.

7.7.2.3. Sample size in regression

The sample size required will depend on the size of effect that we're trying to detect and how much power we want to detect these effects. Specifically, if we want to test the model overall, then a minimum sample size of $50+8k$ is recommended, where k is the number of predictors. If we want to test the individual predictors then a minimum sample size of $104+k$ is recommended.

7.7.2.4. Multicollinearity

Multicollinearity exists when there is a strong correlation between two or more predictors in a regression model. If we have two predictors that are perfectly correlated, then the values of b for each variable are interchangeable. This makes bs untrustworthy.

7.8. How to do multiple regression using R Commander and R

7.8.1. Some things to think about before the analysis

We, not **R**, have to control the quality of the model that is generated

7.8.2. Multiple regression: running the basic model

7.8.2.1. Multiple regression using R commander: the basic model

Album Sales 2.dat. includes the explanatory variables **adverts**, **sales**, **airplay** and **attract**. For the first model, we select **sales** as the response variable, and **adverts** as the explanatory variable. For the second model we choose three explanatory variables, **adverts**, **attract**, and **sales**.

7.8.2.2. Multiple regression using R: the basic model

We can do the same regression using the `lm()` function:

```
album2 <- read.delim("Album Sales 2.dat", header = TRUE)
albumSales.2 <- lm(sales ~ adverts, data = album2)
albumSales.3 <- lm(
    sales ~ adverts +
    airplay +
    attract,
    data = album2
    )
```

7.8.3. Interpreting the basic multiple regression

7.8.3.1. The model summary

To see the output of these models, we execute:

```
summary(albumSales.2)
```

```
Call:
lm(formula = sales ~ adverts, data = album2)

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
```

```
summary(albumSales.3)
```

```
Call:
lm(formula = sales ~ adverts + airplay + attract, data = album2)

Residuals:
     Min       1Q   Median       3Q      Max
-121.324  -28.336   -0.451   28.967  144.132

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -26.612958  17.350001  -1.534    0.127
adverts       0.084885   0.006923  12.261  < 2e-16 ***
airplay       3.367425   0.277771  12.123  < 2e-16 ***
attract      11.086335   2.437849   4.548 9.49e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 47.09 on 196 degrees of freedom
Multiple R-squared:  0.6647,    Adjusted R-squared:  0.6595
F-statistic: 129.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

All of the statistics for `albunSales.2` are the same as the simple regression model earlier (33.5%). However, when the other two predictors are included as well (`albumSales.3`), this value increases to .665. Therefore, we can tell that attractiveness and radio play account for an additional 33.0%

7.8.3.2. Model parameters

The b-values tell us to what degree each predictor affects the outcome **if the effects of all other predictors are held constant**.

- **Advertising budget** ($b = 0.085$): It indicates that as advertising budget increases by one unit, album sales increase by 0.085 units.
- **Airplay** ($b = 3.367$): It indicates that as the number of plays on radio in the week before release increases by one, album sales increase by 3.367 units.
- **Attractiveness** ($b = 11.086$): It indicates that a band rated one unit higher on the attractiveness scale can expect additional album sales of 11.086 units.

In all three factors, t-values are all significant, which indicates all factors are significant predictors of album sales. From the magnitude of the t-statistics we can see that the advertising budget and radio play had a similar impact, whereas the attractiveness of the band had less impact.

To obtain the standardized beta estimates, we need to use a function called `lm.beta()`:

```
lm.beta(albumSales.3)
```

```
  adverts   airplay   attract
0.5108462 0.5119881 0.1916834
```

These estimates tell us the number of standard deviations by which the outcome will change as a result of one standard deviation change in the predictor. The standardized beta values are all measured in standard deviation units and so are directly comparable. The standardized beta values for airplay and advertising budget are virtually identical, indicating that both variables have a comparable degree of importance in the model.

We can also get the confidence intervals by using the `confint()` function.

```
confint(albumSales.3)
```

```
                  2.5 %      97.5 %
(Intercept) -60.82960967  7.60369295
adverts       0.07123166  0.09853799
airplay       2.81962186  3.91522848
attract       6.27855218 15.89411823
```

The confidence intervals of the unstandardized beta values are boundaries constructed such that in 95% of these samples these boundaries will contain the true value of b. A good model will have a small confidence interval, indicating that the value of b in this sample is close to the true value of b in the population. In this model, the two best predictors have very tight confidence intervals, indicating that the estimates for the current model are likely to be representative of the true population values.

7.8.4. Comparing models

7.8.4.1. Comparing models with R Commander

To compare two hierarchical models using R Commander we choose **Models** -> **Hypothesis tests** -> **Compare two models...** . We want to compare `albumSales.2` with `albumSales.3`.

7.8.4.2. Comparing models using R

To compare models using **R** we use the `anova()` function, which takes the general form: `anova(model.1, model.2, ..., model.n)`. The second model must contain everything that was in the first model plus something new, and the third model must contain everything in the second model plus something new, and so on:

```
anova(albumSales.2, albumSales.3)
```

```
Analysis of Variance Table

Model 1: sales ~ adverts
Model 2: sales ~ adverts + airplay + attract
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    198 862264
2    196 434575  2    427690 96.447 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can say that `albumSales.3` significantly improved the fit of the model to the data compared to `albumSales.2`.

7.9. Testing the accuracy of your regression model

7.9.1. Diagnostic tests using R Commander

R Commander allows us to run a range of diagnostic tests and other modifications to our model.

7.9.2. Outliers and influential cases

There are various functions that we can use to obtain different casewise diagnostics.

- **Outliers**
  - Residuals: `resid()`
  - Standardized residuals: `rstandard()`
  - Studentized residuals: `rstudent()`
- **Influential cases**
  - Cook's distances: `cooks.distance()`
  - DFBeta: `dfbeta()`
  - DFFit: `dffits()`
  - Hat values (leverage): `hatvalues()`
  - Covariance ratio: `covratio()`

We can store the values in the dataframe:

```
album2$residuals <- resid(albumSales.3)
album2$standardized.residuals <- rstandard(albumSales.3)
album2$studentized.residuals <- rstudent(albumSales.3)
album2$cooks.distance <- cooks.distance(albumSales.3)
album2$dfbeta <- dfbeta(albumSales.3)
album2$dffit <- dffits(albumSales.3)
album2$leverage <- hatvalues(albumSales.3)
album2$covariance.ratios <- covratio(albumSales.3)
```

In an ordinary sample we would expect 95% of cases to have standardized residuals. Since we have a sample of 200, therefore it is reasonable to expect about 10 cases (5%) to have standardized residuals outside these limits. We can examine it by executing:

```
album2$large.residual <- album2$standardized.residuals > 2 |
  album2$standardized.residuals < -2
sum(album2$large.residual)
```

```
[1] 12
```

Since **R** stores 'TRUE' as 1 and 'FALSE' as 0, we can use the `sum()` function to get the number of cases with a large residual. **R** tells us that only 12 cases had a large residual.

We can also see which cases had TRUE `large.residual`. If we set rows to be `album2$large.residual`, then we will see only those rows for which `large.residual` is TRUE:

```
album2[album2$large.residual,
       c("sales", "airplay", "attract", "adverts", "standardized.residuals")
       ]
```

```
    sales airplay attract  adverts standardized.residuals
1     330      43      10   10.256                2.177404
2     120      28       7  985.685               -2.323083
10    300      40       7  174.093                2.130289
47     40      25       8  102.568               -2.460996
52    190      12       4  405.913                2.099446
55    190      33       8 1542.329               -2.455913
61    300      30       7  579.321                2.104079
68     70      37       7   56.895               -2.363549
100   250       5       7 1000.000                2.095399
164   120      53       8    9.104               -2.628814
169   360      42       8  145.585                3.093333
200   110      20       9  785.694               -2.088044
```

From the output, we can see that we have 12 cases (6%) that are outside of the limits:

```
album2[album2$large.residual ,
       c("cooks.distance", "leverage", "covariance.ratios")
       ]
```

```
    cooks.distance    leverage covariance.ratios
1      0.058703882 0.047190526         0.9712750
2      0.010889432 0.008006536         0.9201832
```

```
10      0.017756472 0.015409738          0.9439200
47      0.024115188 0.015677123          0.9145800
52      0.033159177 0.029213132          0.9599533
55      0.040415897 0.026103520          0.9248580
61      0.005948358 0.005345708          0.9365377
68      0.022288983 0.015708852          0.9236983
100     0.031364021 0.027779409          0.9588774
164     0.070765882 0.039348661          0.9203731
169     0.050867000 0.020821154          0.8532470
200     0.025134553 0.022539842          0.9543502
```

Executing this command prints the variables labelled `cooks.distance`, `leverage`, and `covariance.ratios` but only for cases for which large.residual is TRUE. None of them has a Cook's distance greater than 1, so none of the cases is having an undue influence on the model.

7.9.3. Assessing the assumption of independence

We can test the assumption of independent errors using the Durbin-Watson test. In **R**, we can do this by executing `dwt()` function:

```
dwt(albumSales.3)
```

```
 lag Autocorrelation D-W Statistic p-value
   1       0.0026951      1.949819   0.738
 Alternative hypothesis: rho != 0
```

The closer to 2 that the value is, the better, and the p-value of .7 confirms this conclusion.

7.9.4. Assessing the assumption of no multicollinearity

The VIF and tolerance statistics (with tolerance being 1 divided by the VIF) are useful statistics to access collinearity. We can obtain the VIF using the `vif()` function:

```
vif(albumSales.3)
```

```
 adverts  airplay  attract
1.014593 1.042504 1.038455
```

```
1/vif(albumSales.3)
```

```
  adverts   airplay   attract
0.9856172 0.9592287 0.9629695
```

If VIF values are less than 10 then that indicates there probably isn't cause for concern. If the average of VIF values is not substantially greater than 1, then that also indicates that there's no cause for concern.

7.9.5. Checking assumptions about the residuals

As a final stage in the analysis, we should visually check the assumptions that relate to the residuals. We do this by plotting the standardized residual (y-axis) against the predicted value (x-axis). We can use `plot()` and `hist()` functions to execute this.

From a plot of fitted values against residuals, the funnel shape is typical of heteroscedasticity and indicates increasing variance across the residuals:

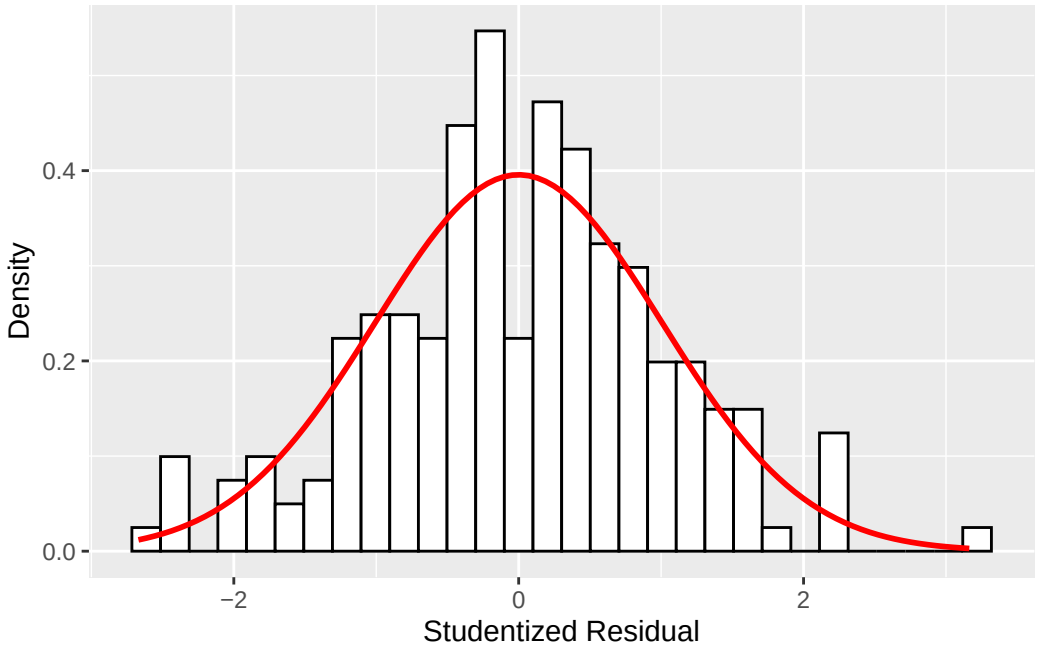
```
album2$fitted <- albumSales.3$fitted.values

histogram <- ggplot(album2, aes(studentized.residuals)) +
  geom_histogram(
    aes(y=..density..),
    color="black", fill="white"
    ) +
  labs(x = "Studentized Residual", y = "Density")
histogram +
  stat_function(
    fun = dnorm,
    args = list(mean = mean(album2$studentized.residuals, na.rm = TRUE),
                sd = sd(album2$studentized.residuals, na.rm = TRUE)
                ),
    color = "red", size = 1
    )
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```

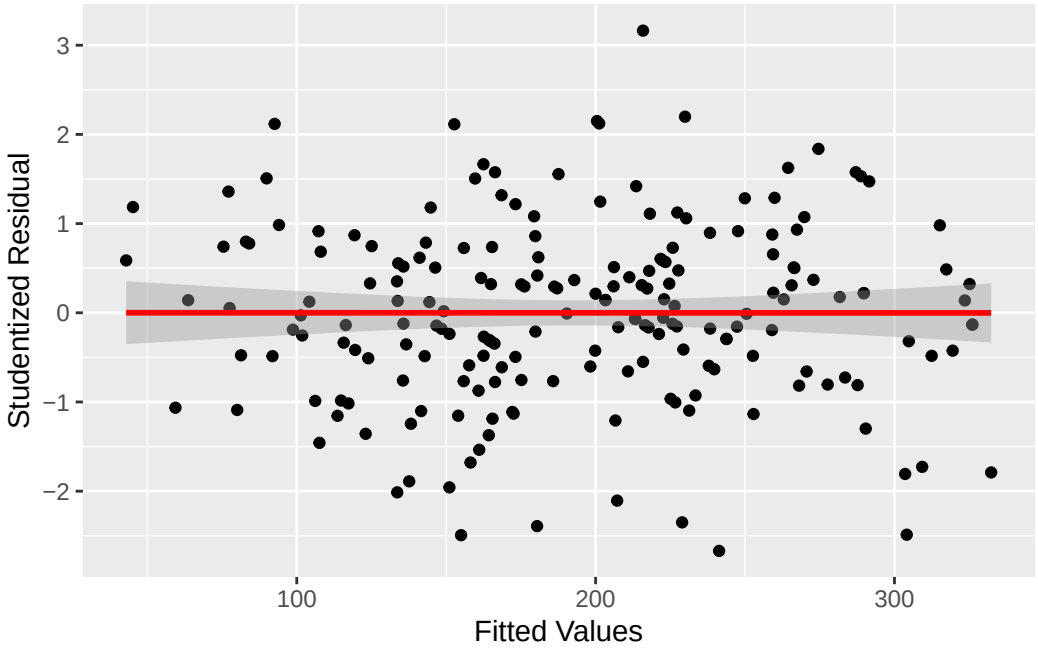
```
Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(density)` instead.
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

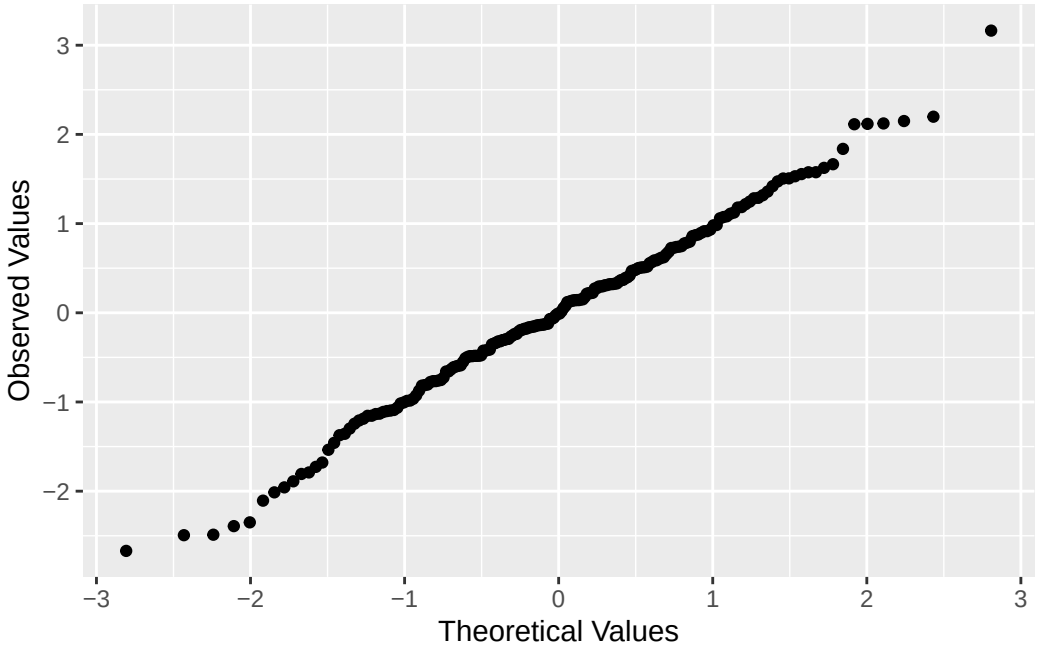



```
scatter <- ggplot(album2, aes(fitted, studentized.residuals))
scatter +
  geom_point() +
  geom_smooth(method = "lm", colour = "Red") +
  labs(x = "Fitted Values", y = "Studentized Residual")
```

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

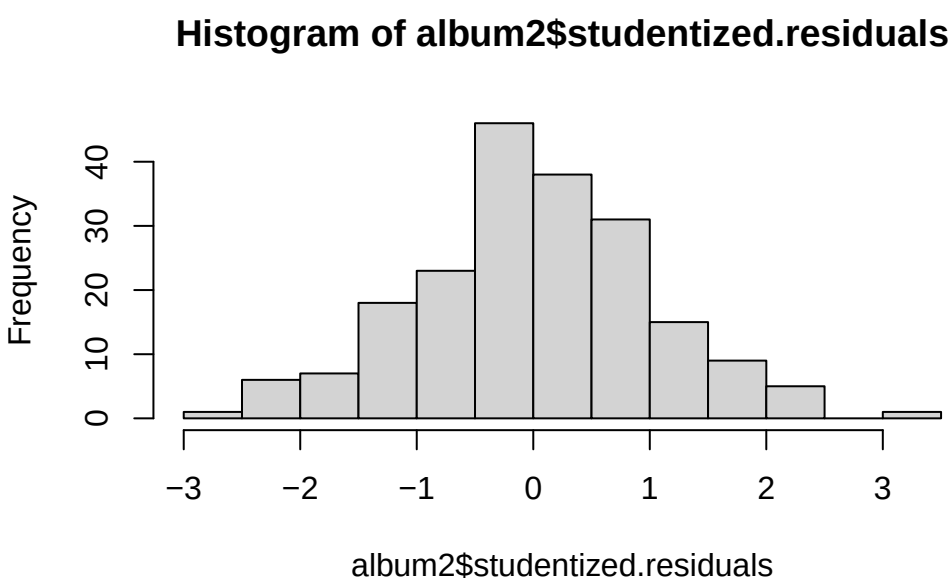



```
qqplot.resid <- ggplot(album2, aes(sample = studentized.residuals)) +
  geom_qq() +  # Use geom_qq to create the QQ plot
  labs(x = "Theoretical Values", y = "Observed Values")
qqplot.resid
```



Another useful way to check whether the residuals deviate from a normal distribution is to inspect the histogram of the residuals by using the `hist()` function:

```
hist(album2$studentized.residuals)
```



The non-normal examples are extreme cases, and we could conclude that in our sample advertising budget and airplay are fairly equally important in predicting album sales.

7.9.6. What if I violate an assumption

If the assumptions have been violated then we cannot generalize our findings beyond our sample.

7.10. Robust regression: bootstrapping

We can bootstrap regression estimates by using the `boot()` function. The locations of the intercept and the coefficients for adverts, airplay and attract are given by `index` values. The type of confidence interval is set as 'bias corrected and accelerated (`type = "bca"`):

```
bootReg<-function(formula, data, i)
{
    d <- data[i,]
    fit <- lm(formula, data = d)
    return(coef(fit))
    }
```

```r
bootResults <- boot(
  statistic = bootReg,
  formula = sales ~ adverts + airplay + attract,
  data = album2, R = 2000
  )

# Confidence interval for the intercept
boot.ci(bootResults, type = "bca", index = 1)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = bootResults, type = "bca", index = 1)

Intervals :
Level       BCa
95%   (-57.77,   6.92 )
Calculations and Intervals on Original Scale
```

```r
# Confidence interval for the coefficient 'adverts'
boot.ci(bootResults, type = "bca", index = 2)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = bootResults, type = "bca", index = 2)

Intervals :
Level       BCa
95%   ( 0.0711,  0.0981 )
Calculations and Intervals on Original Scale
```

```r
# Confidence interval for the coefficient 'airplay'
boot.ci(bootResults, type = "bca", index = 3)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates
```

```
CALL :
boot.ci(boot.out = bootResults, type = "bca", index = 3)

Intervals :
Level       BCa
95%   ( 2.747,  3.926 )
Calculations and Intervals on Original Scale
```

```
# Confidence interval for the coefficient 'attract'
boot.ci(bootResults, type = "bca", index = 4)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = bootResults, type = "bca", index = 4)

Intervals :
Level       BCa
95%   ( 6.54, 15.53 )
Calculations and Intervals on Original Scale
```

Executing this command creates an object called `bootResults` that contains the bootstrap samples. All of the bootstrap confidence intervals are very close to the plug-in confidence intervals, suggesting that we did not have a problem of non-normal distribution in the model.

7.11. How to report multiple regression

7.12. Categorical predictors and multiple regression

7.12.1. Dummy coding

A problem with wanting to use categorical variables as predictors is that often we will have more than two categories. **Dummy variables** are used in these cases. Dummy coding is a way of representing groups of people using only zeros. The number of variables we need is one less than the number of groups we're coding. We need to choose the baseline category in which its dummy variable is assigned as 1. The `contrast()` function is used to set contrasts. The `contr.treatment()` function sets a contrast based on comparing all groups to a baseline condition:

```r
gfr <- read.delim(file = "GlastonburyFestivalRegression.dat", header = TRUE)
gfr$music <- factor(gfr$music)
contrasts(gfr$music) <- contr.treatment(4, base = 4)

crusty_v_NMA <- c(1, 0, 0, 0)
indie_v_NMA <- c(0, 1, 0, 0)
metal_v_NMA <- c(0, 0, 1, 0)
contrasts(gfr$music) <- cbind(crusty_v_NMA, indie_v_NMA, metal_v_NMA)
```

7.12.2. Regression with dummy variables

We can run the regression in the same way as for any other kind of regression, by executing:

```r
glastonburyModel <- lm(change ~ music, data = gfr)
summary(glastonburyModel)
```

```
Call:
lm(formula = change ~ music, data = gfr)

Residuals:
     Min       1Q   Median       3Q      Max
-1.82569 -0.50489  0.05593  0.42430  1.59431

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -0.55431    0.09036  -6.134 1.15e-08 ***
musiccrusty_v_NMA -0.41152    0.16703  -2.464   0.0152 *
musicindie_v_NMA  -0.40998    0.20492  -2.001   0.0477 *
musicmetal_v_NMA   0.02838    0.16033   0.177   0.8598
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6882 on 119 degrees of freedom
  (687 observations deleted due to missingness)
Multiple R-squared:  0.07617,   Adjusted R-squared:  0.05288
F-statistic:  3.27 on 3 and 119 DF,  p-value: 0.02369
```

```r
round(tapply(gfr$change, gfr$music, mean, na.rm = TRUE), 3)
```

```
               Crusty              Indie Kid               Metaller
               -0.966                 -0.964                 -0.526
No Musical Affiliation
               -0.554
```

The result shows that 7.6% of the variance in the change in hygiene can be explained by the musical affiliation of the person.