Applied Research: Advance Statistics with R

Carlos Utrilla Guerrero

6/12/2020

## WELCOME

In workshop #5 you’ll practice constructing and understanding regression models.

### Learning outcomes:

* Create simple and multiple linear regression models with lm()
* Explore the outputs of your models, including coefficients, confidence intervals, and p-values.
* Explore the residuals and fitted values of a linear model.
* Use your model to make predictions for new data

### Essential R assignment(s) document guidelines:

In the current document, you will find the following color(s) highlight(s) and format(s). Please refer to this table for legend description

|  |  |
| --- | --- |
| # this comment: | This is a comment writing by you to describe what you intent to do. |
| print(‘the thing') | This is the thing that you want to run. |
| ## [1] "the output” | ## [1] This is the output of the thing that you run in R. |
| # insert your code here # | This is the expected answer of each question throughout the document. |

**IMPORTANT NOTE:**

If at some point during running your codes you receive the following error message: “...object ‘*variable name*‘ not found then you can solve it by adding the dataset to the code. Here an illustration:

Instead of: name of dataset <- lm(formula= y~x)

You type: name of dataset <- lm(formula = name of dataset$y ~ name of dataset$x)

### Dataset

Throughout the assignments in this workshop, we will use the following dataset:

***Major League Baseball Data***

The baseball.csv file contains *Major League Baseball Data* from the 1986 and 1987 seasons. The data originally comes from an ISLR package. See appendix for a detailed variable description.

<https://docs.google.com/spreadsheets/d/1644rcEM0Jk8qN1l4_2aewfgAuNQ2jg0cJNgSX--ebXM/edit#gid=0>

Using the following command in R, load the baseball.excel data into R and store it as a new object called hitters.

library(readxl) # import library  
baseball <- read\_excel("Baseball.xlsx") # import the baseball.xlsx file

\*Note that this command assumes that you have the readxl package already installed. If you haven’t done this in previous meetings but imported the file differently, then you first have to run the command install.packages(“readxl”).

Firs you want to get to know the data set and see whether importing the dataset was successful and worked out correctly. To do this, first take a look at the *first 6 rows* of the dataset(s) by printing it/them to the console. Which command allows you to do this? The output should look like the table below:

## # A tibble: 6 x 20  
## AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns CRBI  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 215 51 4 19 18 11 1 215 51 4 19 18  
## 2 458 114 13 67 57 48 4 1350 298 28 160 123  
## 3 586 159 12 72 79 53 9 3082 880 83 363 477  
## 4 568 158 20 89 75 73 15 8068 2273 177 1045 993  
## 5 399 102 3 56 34 34 5 670 167 4 89 48  
## 6 579 174 7 67 78 58 6 3053 880 32 366 337  
## # ... with 8 more variables: CWalks <dbl>, League <chr>, Division <chr>,  
## # PutOuts <dbl>, Assists <dbl>, Errors <dbl>, Salary <dbl>, NewLeague <chr>

Next, you take a look at the structure of the data in order to see the number of observations the dataset *baseball* has, as well as the number *and* type of variables. Which code can you use? The output should look like...or give you exactly this information!

## tibble [50 x 20] (S3: tbl\_df/tbl/data.frame)  
## $ AtBat : num [1:50] 215 458 586 568 399 579 496 510 551 313 ...  
## $ Hits : num [1:50] 51 114 159 158 102 174 119 126 171 78 ...  
## $ HmRun : num [1:50] 4 13 12 20 3 7 8 2 13 6 ...  
## $ Runs : num [1:50] 19 67 72 89 56 67 57 42 94 32 ...  
## $ RBI : num [1:50] 18 57 79 75 34 78 33 44 83 41 ...  
## $ Walks : num [1:50] 11 48 53 73 34 58 21 35 94 12 ...  
## $ Years : num [1:50] 1 4 9 15 5 6 7 11 13 12 ...  
## $ CAtBat : num [1:50] 215 1350 3082 8068 670 ...  
## $ CHits : num [1:50] 51 298 880 2273 167 ...  
## $ CHmRun : num [1:50] 4 28 83 177 4 32 36 44 128 35 ...  
## $ CRuns : num [1:50] 19 160 363 1045 89 ...  
## $ CRBI : num [1:50] 18 123 477 993 48 337 280 519 900 321 ...  
## $ CWalks : num [1:50] 11 122 295 732 54 218 165 256 917 170 ...  
## $ League : chr [1:50] "A" "A" "N" "N" ...  
## $ Division : chr [1:50] "E" "W" "E" "W" ...  
## $ PutOuts : num [1:50] 116 246 181 105 211 ...  
## $ Assists : num [1:50] 5 389 13 290 9 479 371 358 149 206 ...  
## $ Errors : num [1:50] 12 18 4 10 3 5 29 20 5 7 ...  
## $ Salary : num [1:50] 70 475 1043 775 80 ...  
## $ NewLeague: chr [1:50] "A" "A" "N" "N" ...

Now, you see that this data contains 50 observations and 20 variables. Also, we can clearly see that most of them are numerical and few of them are characters (characters = text/string). Furthermore, none of the numerical variables are categorical ones, while all of the character variables are categorical. However, since the categorical variables are considered characters, they can’t be effectively used in further analysis. Therefore, we need to convert them into factors.

\*Note, due to the fact that we import the file from excel into R, R does not directly recognize categorical variables as factor and therefore we need to convert them ourselves.

### Exercise 1: Can you please convert all three categorical variables into a factor? Remember from workshop 2 the code?

### Check whether you have successfully converted the variables by using the code class(baseball$variable you want to check)

Does the code class tell you that all the 3 categorical variables have been changed into a factor? If yes, then perfect! Another way to check it is having a look again the structure of our baseball dataset. The difference to class() is that the next command also immediately tells you how many levels the individual factor variables have.

str(baseball)

## tibble [50 x 20] (S3: tbl\_df/tbl/data.frame)  
## $ AtBat : num [1:50] 215 458 586 568 399 579 496 510 551 313 ...  
## $ Hits : num [1:50] 51 114 159 158 102 174 119 126 171 78 ...  
## $ HmRun : num [1:50] 4 13 12 20 3 7 8 2 13 6 ...  
## $ Runs : num [1:50] 19 67 72 89 56 67 57 42 94 32 ...  
## $ RBI : num [1:50] 18 57 79 75 34 78 33 44 83 41 ...  
## $ Walks : num [1:50] 11 48 53 73 34 58 21 35 94 12 ...  
## $ Years : num [1:50] 1 4 9 15 5 6 7 11 13 12 ...  
## $ CAtBat : num [1:50] 215 1350 3082 8068 670 ...  
## $ CHits : num [1:50] 51 298 880 2273 167 ...  
## $ CHmRun : num [1:50] 4 28 83 177 4 32 36 44 128 35 ...  
## $ CRuns : num [1:50] 19 160 363 1045 89 ...  
## $ CRBI : num [1:50] 18 123 477 993 48 337 280 519 900 321 ...  
## $ CWalks : num [1:50] 11 122 295 732 54 218 165 256 917 170 ...  
## $ League : Factor w/ 2 levels "A","N": 1 1 2 2 1 2 2 2 2 2 ...  
## $ Division : Factor w/ 2 levels "E","W": 1 2 1 2 2 1 2 2 1 2 ...  
## $ PutOuts : num [1:50] 116 246 181 105 211 ...  
## $ Assists : num [1:50] 5 389 13 290 9 479 371 358 149 206 ...  
## $ Errors : num [1:50] 12 18 4 10 3 5 29 20 5 7 ...  
## $ Salary : num [1:50] 70 475 1043 775 80 ...  
## $ NewLeague: Factor w/ 2 levels "A","N": 1 2 1 2 2 1 2 2 1 2 ...

Great. It seems like we are ready to run some basic summary statistics of our dataset now, in order to get a first impression of the data distribution. Which code can you use? The output should look like...or give you exactly this information!

## AtBat Hits HmRun Runs   
## Min. : 19.0 Min. : 1.0 Min. : 0.00 Min. : 0.00   
## 1st Qu.:288.0 1st Qu.: 77.0 1st Qu.: 5.25 1st Qu.: 35.75   
## Median :418.5 Median :105.0 Median :10.00 Median : 52.50   
## Mean :407.6 Mean :108.3 Mean :11.04 Mean : 54.14   
## 3rd Qu.:541.8 3rd Qu.:147.2 3rd Qu.:15.50 3rd Qu.: 67.75   
## Max. :642.0 Max. :211.0 Max. :31.00 Max. :107.00   
## RBI Walks Years CAtBat   
## Min. : 0.00 Min. : 0.00 Min. : 1.00 Min. : 19.0   
## 1st Qu.: 33.00 1st Qu.:26.25 1st Qu.: 3.25 1st Qu.: 806.2   
## Median : 46.00 Median :34.50 Median : 6.00 Median :1993.0   
## Mean : 50.60 Mean :40.66 Mean : 7.10 Mean :2767.2   
## 3rd Qu.: 69.75 3rd Qu.:56.75 3rd Qu.:11.00 3rd Qu.:4730.2   
## Max. :116.00 Max. :94.00 Max. :16.00 Max. :8068.0   
## CHits CHmRun CRuns CRBI   
## Min. : 4 Min. : 1.00 Min. : 2.0 Min. : 3.0   
## 1st Qu.: 202 1st Qu.: 18.75 1st Qu.: 100.5 1st Qu.: 75.0   
## Median : 542 Median : 38.00 Median : 277.0 Median : 264.0   
## Mean : 762 Mean : 63.58 Mean : 368.6 Mean : 335.6   
## 3rd Qu.:1282 3rd Qu.: 83.00 3rd Qu.: 581.8 3rd Qu.: 475.8   
## Max. :2273 Max. :305.00 Max. :1135.0 Max. :1234.0

## CWalks League Division PutOuts Assists   
## Min. : 1.0 A:25 E:24 Min. : 0.0 Min. : 0.0   
## 1st Qu.: 65.0 N:25 W:26 1st Qu.: 116.0 1st Qu.: 7.0   
## Median :163.0 Median : 225.5 Median : 46.5   
## Mean :253.3 Mean : 286.5 Mean :130.9   
## 3rd Qu.:380.5 3rd Qu.: 312.0 3rd Qu.:216.5   
## Max. :917.0 Max. :1236.0 Max. :487.0   
## Errors Salary NewLeague  
## Min. : 0.00 Min. : 70.0 A:26   
## 1st Qu.: 5.00 1st Qu.: 167.5 N:24   
## Median : 7.50 Median : 542.5   
## Mean : 9.54 Mean : 608.8   
## 3rd Qu.:12.75 3rd Qu.: 768.8   
## Max. :29.00 Max. :2127.3

Next thing to do is to check for missing values. We do this with the command is.na(). If you run it, you should notice that there aren’t any NA values in our dataset; that is perfect! Good to know: summary()also will give you a count of NA values (provided they have already been coded in R as NA; a value like -99 isn’t automatically recognized as NA unless you have told R that it is an NA).

At this point, we are very interested inn exploring the Baseball dataset in more detail. We are wondering if there is a relationship between the number of Hits a player had and his Salary. Lets create a scatterplot for this. Try this command in your R console

library(ggplot2) # import ggplot2 library for visualisation  
ggplot(baseball, aes(x = Hits, y = Salary)) +  
 geom\_point(col = "pink") +  
 labs(title = "Is there any relationship between baseball player Hits vs Salary",  
 subtitle = "Plotting your data is the first step to figuring out",  
 caption = "R course Venlo Course")

\*Note again, that library() only works if you have already installed the related package previously, in this case install.packages(“ggplot2”).

A screenshot of a cell phone

Description automatically generated

The y-axis is the amount of salary ***(the dependent variable is always on the y-axis)*** and the x-axis is the total Hits. Each pink dot represents one player the baseball dataset. Glancing at this data, you probably notice that salary are higher for players that hit a lot. That’s interesting to know, but by how much is the salary higher?

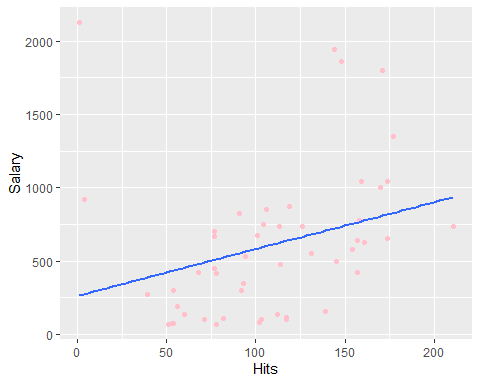
How can we answer with some degree of certainty how much a player typically earns when he hits a certain amount of balls? We can do this by drawing a line through the chart above, one that runs roughly through the middle of all the data points. We do this as follows:

ggplot(baseball, aes(x = Hits, y = Salary)) +  
 geom\_point(col = "pink") +  
 geom\_smooth(method='lm', # Add linear regression line   
 se = FALSE) # # Don't add shaded confidence region

labs(title = "Is there any relationship between baseball player Hits vs Salary",  
 subtitle = "Plotting your data is the first step to figuring out",  
 caption = "R course Venlo Course")

##You might get the error geom\_smooth()`using formula 'y ~ x'. Ignore this error. Despite this error you get the correct graph in the plots window.

The resulting blue line (see plot below) is called the regression line and it’s the line that best fits the data. In other words, the blue line is the best explanation of the relationship between the independent variable (Hits) and dependent variable (Salary).



## Exercise 2: Based on this plot, do you expect there to be a relationship between these variables? Which kind of test would you run in order to quantitatively support your answer? Run this test and interpret the results

\*Have a look at workshop 4 if you are wondering which test would be appropriate.

In addition to drawing the line you can use the information in the scatterplot to calculate the slope and y-intercept of this best-fitting regression line. Then you can use the following statistical formula to interpret any value you like:

Salary=B0+B1\*Hits

In this case, B0 is the intercept, the value from where you start measuring (or in the figure the value where the regression line starts when x (in this case the Hits) = 0). B1 is the slope of the best-fitting line. The slope measures the change of Salary with respect to the Hits as predicted by the line. Stated differently, the line illustrates the salary increase (B1) for every one more hits a player did achieve it. To fully understand what we are talking about, we need to introduce you to what a ‘Simple Linear Regression’ is.

### Exercise 3: Can you fill in the formula based on the scatterplot and best-fitting line you have created in R? As it is difficult to derive the exact numbers visually from a scatterplot, approximate numbers will do!

## SIMPLE LINEAR REGRESSION

### What is a linear regression?

A linear regression is a statistical model that analyzes the relationship between a dependent variable (y) and one or more other variables (often explanatory variables or independent x). You make this kind of relationships in your head all the time, for example when you calculate the age of a child based on her height, you are assuming the older she is, the taller she will be. Or just the example taken above as a way to hypothesize an expected relationship between Salary and Hits.

As mentioned before, we attempt to create a linear model to see this relationship. This model will predict Salary, using, in our case, the explanatory variable hits.

Using the guide below, we are going to conduct a simple linear regression analysis predicting a player’s Salary as a function of his Hits value. We are going to save the result of this analysis to a new object in R called baseball\_simple:

baseball\_simple <- lm(formula = Salary ~ Hits, # linear regression formula (Dependent ~ Independent)  
 data = baseball) # the name of my dataset

With the command summary(newly created object), thus in this case summary(baseball\_simple) you get detailed information on how well the model predicts salary (the dependent variable) and the coefficients of the model.

summary(baseball\_simple)

##   
## Call:  
## lm(formula = Salary ~ Hits, data = baseball)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -546.9 -337.7 -127.9 183.6 1861.2   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 262.886 176.440 1.490 0.1428   
## Hits 3.195 1.499 2.131 0.0382 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 489.6 on 48 degrees of freedom  
## Multiple R-squared: 0.08646, Adjusted R-squared: 0.06743   
## F-statistic: 4.543 on 1 and 48 DF, p-value: 0.0382

### Interpretation

R will give you always the following information if you ask for the output of the simple linear regression model:

***Call***

## Call:  
## lm(formula = Salary ~ Hits, data = baseball)  
##   
This tells you what you actually just run. The linear equation of:

lm(formula = Salary ~ Hits, data = baseball)

***Residuals***

## Residuals:  
## Min 1Q Median 3Q Max   
## -546.9 -337.7 -127.9 183.6 1861.2   
##

Residuals are essentially the difference between the observed values (salary) and the response values that the model predicted. The Residuals section of the model output breaks it down into 5 summary points. Later on when making a residual plot, you are going to see that the distribution of the residuals do not appear to be strongly symmetrical. That means that the model gives you certain *predictions* that fall far away from the *actual* observed points.

We could take this further consider plotting the residuals to see whether this normally distributed, etc. but will skip this for this example for the moment.

***Coefficients***

## **Estimate** Std. Error t value Pr(>|t|)   
## **(Intercept) 262.886** 176.440 1.490 0.1428   
## **Hits 3.195** 1.499 2.131 0.0382 \*

This part of the output gives us our *estimated* coefficients (betas). The first one given is 262.886, which is our estimated intercept value (B0). This is the average value for Salary when Hits are zero. What it is saying statistically speaking is that when Hits are zero, we would expect an increase of the players salary of 262.886 dollar on average. However, this information usually means nothing in real life. The next estimated coefficient 3.19 corresponds to B1, the slope of the best fit regression line. This value is really important! It shows us the numerical relationship between our independent and dependent variable. It tells us that a one unit increase in the independent variable x (in our case an increase of Hits), there will be an expected change in Salary (dependent variable y), on average of 3.195units. And this has a practical meaning! For one Hits increase that baseball players does, we expect them to win 3.195 more dollars in their salary on average. Note that we can interpret the model this way because we have two continuous variables.

To come back to the formula we asked you to fill in in Exercise 3. Now we have the exact values that we can put in our mathematical equation and do no longer have to rely on approximate values:

Salary=B0+B1\*Hits

🡪 Salary=262.886+3.195Hits

**This is the trickiest part to get when interpreting your model results so take your time to fully understand it!**

But what do the other coefficients mentioned in the output mean?

## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 262.886 176.440 1.490 0.1428   
## Hits 3.195 1.499 2.131 0.0382 \*

***Std. Error***

Since we have an estimate for a coefficient, it has some errors attached to it. Remember that standard error is the variability we would expect in the variable Hits when we would conduct repeated sampling (lecture workshop 4). So, this column captures the sampling variability. For instance, we might have estimated value for Hits of e.g. 3.100 or 3.197 and so on, so this variability is entirely captured by the standard error.

***t value***

The t value of our ‘Estimate’ is just the ratio of the ‘Estimate’ divided by the standard error. In other words, how big is my estimate relative to its error. The bigger the difference between them, the bigger the t value. And that implies the bigger the t value the better ☺.

*p-value*

Each t value has it is own p value. This p value is just telling us, as you know, how statistically significant each of our estimates is in our model. In our model, we do not care about the intercept. Rather, we care about p values for our slope (the variable Hits). For the independent variable Hits, the p value in our model is quite small 0.0382. Therefore, we will reject the null hypothesis (e.g. Hits estimate is equal to zero) and say informally speaking: “yes, this 3.19 increase is statistically significant at the 0.05 level”. This also means that we expect there to be a positive relationship between hits and salary in all players.

***Signif. Codes***

## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The p-value is followed in this output/case by a \*. The signif. codes tell us that (\*) is associated with the level of significance of 5%. Thus the p-value is significant at the level of significance of 5%. Not e.g. 1% because then the p-value in the output would have been followed by (\*\*).

***Residual standard error***

## Residual standard error: 489.6 on 48 degrees of freedom

It is a very basic quantification of how well or poorly our model is doing at predicting our data on average. You can think of it like “the average error for your model”. Sometimes a model really predicts closely the data and sometimes it does not. We want our model to have a small residual error. In our example, the model when it tries to use Hits variable to predict the number of increase Salary is off on an average of about 489.6 (e.g. either on average predict 489.6 Hits to high or low when predicting individual data points).

***Multiple R-Squared and Adjusted R-squared***

## Multiple R-squared: 0.08646, Adjusted R-squared: 0.06743

These metrics are used to assess the model accuracy.

The Multiple R-squared (Standard R squared) is interpreted as the percentage of the variation in the response variable (Y) that is explained by the variation in the explanatory variable (X). In other words, it determines how well our model is doing at explaining the total variation of the response variable (Y). It always lies between 0 and 1 (i.e.: a number near 0 represents a regression that does not explain the variance in the response variable well and a number close to 1 does explain the observed variance in the response variable). Nevertheless, it’s hard to define what level of R2 is appropriate to claim the model fits well. Essentially, it will vary with the application and what is considered well in a field of study.

The adjusted R-squared is similar to the Multiple R-squared but with an adjustment: the number of variables adding into the model. This adjusted R-squared takes into account how many explanatory(X) variables are in your model. What it does is penalize you for adding useless (e.g. not statistically significant) variables to the model. In our example, the R2 we get is 0.06743. Or roughly 6.743% of the variance found in the response variable (*Salary*) can be explained by the explanatory variable (*Hit*). Step back and think: If you were able to choose any variable in the baseball dataset to predict an increase of players *Salary*, would number of *Hit* be one and would it be an important one that could help explain how *Salary* would vary based on *Salary*? I guess it’s easy to see that the answer would almost certainly be a yes. However, the fact that the model only explains approx.. 7% of the variation in *Salary*, it seems that there are many more variables that might be of importance to predict *Salary* (e.g. Walks, Runs, CAtBat).

*Note*: Later on we will discuss multiple regression. In multiple regression you try to predict the dependent variable y using *more than one* explanatory/independent variable. In multiple regression settings, the Multiple R-squared will always be higher to the simple regression one, because of the fact that more variables are included in the model. However, if the variables you are adding are not statistically significant, then adjusted R-squared takes this into account, and presents a much lower value than Multiple R-squared. Therefore, the adjusted R squared metric is a good measure when starting to consider more complex models that include multiple independent variables.

***F-statistic***

## F-statistic: 4.543 on 1 and 48 DF, p-value: 0.0382

F is a ratio of how well the model is doing over the errors (e.g. mean square model / mean square errror). A higher F values means that we have a better model.

F-Statistic measures the significance of the overall model. In our case, since we only included one variable in our model (*Hit*) which turned out to be statistically significant at 5% (\*), the F-Statistic will be also be statistically significant. Thus, when adding more variables (e.g. multiple regression) this situation might change, and we will be able to distinguish between statistically significance of individual variables versus statistically significance of the overall model.

The p value for F statistic indicates that the overall model is significant. Or stated differently that in this simple regression model that contains two variables (one dependent and one independent), the relationship between these two variables is significant. But we have seen that the R squared is low (=0.06743). So can a regression model that is significant but has a small R-squared be useful?

Thus, how can be it possible? How can be that our model is significant with a R squared of 0.06743? Can a Regression Model with a Small R-squared Be Useful? It’s easy to dismiss the model as being useless. You’re only explaining approx. 6% of the variation, Why bother? If the only point of our model is prediction, the model would do a pretty bad job. But it isn’t.

Think about this...If you consider all of the things that might affect a player’s salary, do you really expect performance to be the major and unique contributor? Even though I’m not a sport researcher, I can think of quite a few variables that I would expect to be much better predictors of salary. Things like age, error history, assist, League division could also help us to determine how much worth a player is, isn’t? And putting all of them into the model would indeed give better predicted values.

The first focal point was to see if there was a small, but reliable relationship between `Salary` and `hits` variable. And there was. In order to create a much more powerful model prediction, we will just need to consider adding other variables in order to improve its prediction ability. We will learn this later on when discussing Multiple linear Regression.

***Predicted values estimated by our Linear Model (Fitted Values):***

As discussed you can use the regression formula ‘Salary=262.886+3.195Hits’ to calculate the salary of each of our player incorporated in the baseball dataset by hand. However, we can also ask R to do it for us.

If you type in R baseball\_simple$fitted.value, you will see all your estimated/fitted values for each player/ observation:

# estimated values  
baseball\_simple$fitted.values

## 1 2 3 4 5 6 7 8   
## 425.8066 627.0619 770.8157 767.6212 588.7276 818.7336 643.0345 665.3962   
## 9 10 11 12 13 14 15 16   
## 809.1500 512.0589 764.4266 620.6728 512.0589 595.1166 441.7793 681.3689   
## 17 18 19 20 21 22 23 24   
## 432.1957 508.8644 563.1713 489.6972 556.7823 266.0802 764.4266 559.9768   
## 25 26 27 28 29 30 31 32   
## 435.3902 585.5330 722.8978 636.6455 275.6638 818.7336 387.4723 623.8674   
## 33 34 35 36 37 38 39 40   
## 805.9555 636.6455 435.3902 726.0923 480.1136 777.2047 706.9251 601.5057   
## 41 42 43 44 45 46 47 48   
## 735.6759 508.8644 454.5574 936.9312 553.5877 591.9221 508.8644 754.8430   
## 49 50   
## 828.3172 524.8370

This list of numbers are basically the new salary amounts per player that our model estimate. For instance, the first player will earn 425.8066 dollar according to our linear model.

# Predicted value for first player  
head(baseball\_simple$fitted.values,1)

## 1   
## 425.8066

If we look at the observed salary that this player got, we can see that the differences between them is quite large:

# Salary for the first player  
head(baseball$Salary,1)

## [1] 70

Let’s check how many hits he had according to our datafile:

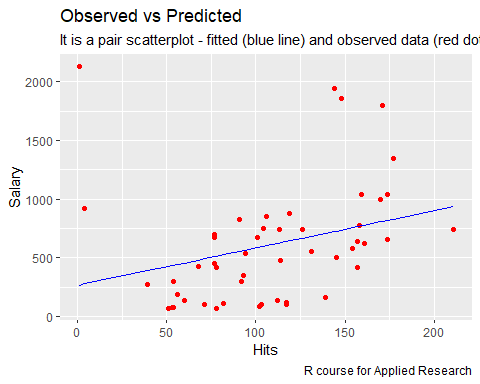
head(baseball$Hits,1)

## [1] 51

Wow! That difference between this two numbers is too large! 355.806613 dollar more!! Our model did not work very well for predicting purposes when it comes down to observation/player #1!

What we just have done here is to look at the difference between the observed (Salary) and the estimated value of Salary (the fitted value). These differences are called: Residuals (or errors). We will come back to them in a later stage. Now, lets generate a plot that shows us the differences for each observation. This will give us a visual sense of what the variation between observed and fitted values is.

ggplot() +  
 geom\_point(aes(x = baseball$Hits, y = baseball$Salary),  
 colour = 'red') +  
 geom\_line(aes(x = baseball$Hits, y = fitted(baseball\_simple)),  
 colour = 'blue') +  
 labs(title = "Observed vs Predicted", x = "Hits", y= "Salary",  
 subtitle = "It is a pair scatterplot - fitted (blue line) and observed data (red dots)",  
 caption = "R course for Applied Research")



We can see that for the above example (observation #1; the player that made 51 hits), the observed salary was 70 whereas our model predicted a salary of roughly 425. You can see it on the graph in yellow (yellow dot). So this plot is telling you the differences between the observed salary (red dots) and predicted salary (blue line). Ideally, we would like the red dots close to the blue line to conclude that our model predict Salary well. But it is not the case.

Another interesting form of data visualization that helps you assess your model performance is to create a scatter plot of Observed vs Fitted. You can tell pretty much everything from it. It is a scatter plot with observed values in the x-axis and fitted (or predicted) values in y-axis. We will create a plot like this and also add a diagonal line on it. Try out this code:

# Create scatterplot for observed vs fitted   
ggplot(data = baseball, aes(x = Salary, y = baseball\_simple$fitted.values)) +   
 geom\_point() +  
 geom\_abline(slope = 1, intercept = 0, col = "red") +  
 labs(title = "Relationship between model fits (predicted) and observed Salaries",  
 subtitle = "Simple Regression = Salary ~ Hits",  
 caption = "R course for Applied Research", y = 'Predicted (Fitted) Salary', x = 'Observed Salary') +  
 geom\_segment(aes(x = Salary, y = baseball\_simple$fitted.values, xend = Salary, yend = baseball\_simple$fitted.values),   
 col = "red") +  
 xlim(c(-300, 3000)) +  
 ylim(c(-300, 3000))

A close up of a map

Description automatically generated

Ideally, all your points should be close to the regressed diagonal line (the red line). For perfect prediction, you would have fitted=observed, or x=y. So when you draw that red line through this graph above, you see how much the prediction deviates from the observed value (the residuals). To some extent, in the graph, you can note that the predictions was mostly overestimating the observed outcome (predicted>observed). So, for instance taking the first row, the observed salary is 70, your predicted should be reasonably close to 70 to. If the Salary equals 200, your predicted should also be reasonably close to 200. If your model had a high R Square, all the points would be close to this diagonal line. The lower the R Square, the weaker the Goodness of fit of your model, the more foggy or dispersed your points are (away from this diagonal line).

You will have to conclude that our model seems to have clearly no relationship between the model’s fitted values and Observed ones.

## 

## Residual Plot

A linear regression model is not always the right approach to modeling the relationship between your variables. E.g. when your variables are related in a U-curve then linear regression is not the appropriate method to use. To check this, you should assess the appropriateness of the model by defining residuals and examining residual plots.

### What are the residuals?

The difference between the observed value of the dependent variable (Salary) and the predicted value (fitted values) is called the residual (e). Each data point has one residual.

Residual(e) = Observed value - Fitted value

A residual plot is a graph that shows the residuals on the vertical axis and the independent variable (in this case Hits) on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, a nonlinear model is more appropriate. e = y - ŷ

The residual data of the simple linear regression model is the difference between the observed data and fitted values of the dependent variable.

We can create a table that contains the y = Observed Values, ŷ = fitted values and e = residual(y-ŷ) using the following code:

# add the error (residuals = Observed - Fitted)  
residuals <- baseball$Salary-fitted(baseball\_simple)  
  
# create residual table  
residual\_table <- data.frame(Hits = baseball$Hits, Observed = baseball$Salary,Fitted = fitted(baseball\_simple), residuals)  
  
# print first player predicted value and observed salary  
head(residual\_table,1)

## Hits Observed Fitted residuals  
## 1 51 70 425.8066 -355.8066

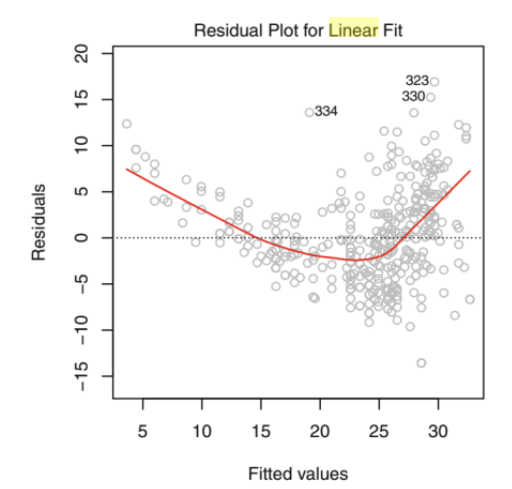
From the above table for observations/player 1, you can see that the salary was 70 dollars (Observed column) and our model (baseball\_simple) estimated a salary of 425.8066 (Fitted column) with an residual (residuals) of -355.066. Now that we grasp the concept of residuals let’s plot the residuals and fitted values as a residual plot. Here we have our residual plot:

# plot the residuals against the fitted values  
plot(baseball\_simple,1)

A close up of a map

Description automatically generated

Ideally, the residual plot will show no fitted pattern. That is, the red line should be approximately horizontal at zero. The presence of a pattern may indicate a problem with some aspect of the linear model. In our example, there is no pattern in the residual plot. This suggests that we can assume linear relationship between the *Salary* and the *Hits* variables. In the figure below you see an illustration of a residual plot that shows a strong pattern in the residuals due to the non-linearity of the data.

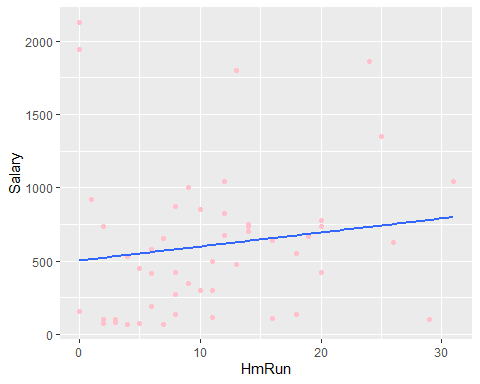


To conclude, you have seen a simple linear regression that has confirmed a statistically significant relationship between Salary and Hits. Also, we could conclude that our model is statistically significant with a F-value that confirms the overall goodness of fit of our model. Indeed, the analysis of the residuals also has shown no presence of linearity and thus, we might assume the appropriateness of the linear regression. Nonetheless, our model has a low R-squared which indicates that its predictive ability is quite weak and we need to further investigate which other explanatory variables could be usefull to add into the model and help us to explain more amount of variance of the Salary.

### Exercise 4 – Now it is your turn: You are asked to answer the question: “Is there a relationship between the number of home runs a player scores and salary” using simple linear regression.

1. Create a scatterplot showing the relationship (including a line) between HmRun and Salary using the code just learned. The output should look like this:

Based on this plot, what do you expect the relationship to be between HmRun and Salary?



1. Conduct a regression analysis predicting a player’s Salary as a function of its HmRun value. Save the result to an object called baseball\_simple1 and ask R to give you the output (which should look like below). Interpret the output.

## Call:  
## lm(formula = Salary ~ HmRun, data = baseball)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -680.3 -363.8 -104.8 183.5 1624.0   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 503.340 125.280 4.018 0.000206 \*\*\*  
## HmRun 9.552 9.308 1.026 0.309950   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 506.7 on 48 degrees of freedom  
## Multiple R-squared: 0.02147, Adjusted R-squared: 0.001081   
## F-statistic: 1.053 on 1 and 48 DF, p-value: 0.3099

1. Interpret the quality of the model based on your baseball\_simple1 model result please.
2. Plot the relationship between the model fitted values and the observed values.

A nice tutorial for a simple regression is here: <https://www.datacamp.com/community/tutorials/linear-regression-R>

## Multiple Regression

Multiple linear regression is an extension of simple linear regression. It is used to predict a dependent variable (y) on the basis of multiple explanatory/independent variables (x).

When including four predictor variables (x), the prediction of y is expressed by the following equation:

Salary (y)=B0+B1\*x1 + B2\*x2 + B3\*x3 + B4\*x4

The “B” values are called the regression weights (or beta coefficients). They measure the association between the explanatory variables and the dependent variable.

In this chapter, you will learn how to:

Build and interpret a multiple linear regression model in R, in addition to checking the overall quality of the model.

## Building the model

We want to create a model for estimating the players salary based on the following variables from the baseball file:

1. Hits - Number of hits in year
2. CWalks - Number of walks during his career
3. Assists - Number of assists in year
4. Errors - Number of errors in year

To do this, use the formula = ‘y ~ x1 + x2 + x3 + Errors’ notation. Assign the result to an object called baseball\_multiple.

baseball\_multiple <- lm(formula = Salary ~ Hits + CWalks + Assists + Errors, # linear regression formula (Dependent ~ Independent)  
 data = baseball) # the name of my dataset

Again, create a summary() baseball\_multiple in order to check the results of the regression:

summary(baseball\_multiple)

##   
## Call:  
## lm(formula = Salary ~ Hits + CWalks + Assists + Errors, data = baseball)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -800.2 -232.2 -135.6 203.1 1681.5   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 288.4273 169.9846 1.697 0.096645 .   
## Hits 0.1785 1.4668 0.122 0.903676   
## CWalks 1.0529 0.2859 3.683 0.000616 \*\*\*  
## Assists 0.9540 0.4906 1.945 0.058069 .   
## Errors -9.4864 10.8338 -0.876 0.385883   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 420.3 on 45 degrees of freedom  
## Multiple R-squared: 0.3688, Adjusted R-squared: 0.3127   
## F-statistic: 6.573 on 4 and 45 DF, p-value: 0.0002967

### 

### Interpretation

The first step in interpreting the multiple regression analysis is to examine the F-statistic and the associated p-value, at the bottom of model summary.

In our example, it can be seen that p-value of the F-statistic is < 0.0002967, which is highly significant. This means that, at least, one of the predictor variables is significantly related to the Salary variable.

To see which explanatory variables are significant, you can examine the coefficients table, which shows the estimate of regression beta coefficients and the associated t-statitic and p-values. For a given predictor, the t-statistic evaluates whether or not there is a significant association between a specific explanatory variable and the dependent variable, that is whether the beta coefficient of the predictor is significantly different from zero.

It can be seen that, changing in Number of walks during his career (CWalks) is significantly associated to changes in salary while changes in Assist is not statistically significant related to Salary at the 5% level, but only at the 10% level of significance.

As you might remember from the simple regression explanation, for a given explanatory variable, the coefficient (B) can be interpreted as the average effect on dependent variable (y) of a one unit increase in the explanatory variable, holding all other variables fixed.

For example, making an additional 100 walk in one player carrer CWalks leads to an increase in salary by approximately 1.0528\*100 = 105.28 dollars, on average.

The Assists coefficient suggests that for every 1 run increase in one player career, holding all other explanatory variables constant, we can expect an increase of 0.9540434\*100 = 95.40 dollars, on average.

In constrast, the only negative relationship is with Errors in player career. Indeed, it is notoriously penalized to make errors in the Baseball League. One additional Error in one player career, holding all other explanatory variables contant, we can expect an decrease of -9.486\*100 = -948.6.

Regarding p-values, they tell you in this model whether the hypothesis that a specific predictor is not meaningful for your model can be rejected. The p-value for CWalks is 0.000616. A very small value means that CWalks is seen as an excellent addition to your model. What becomes interesting is to see how our previous variable that we added in the simple linear regression before, is now, not statistically significant. It may be possible since there might be some problem of multicolinearity (i.e. different variables that mean qualitatively the same). But this claim, needs to be further analyzed and confirmed and for now, we will not deeply investigate how to do this here. The last variable that is also not significant is the variable Error with a p-value of 0.385883. In other words, there’s 38% chance that this explanatory variable is not meaningful for the regression model.

### 

### Model accuracy assessment

As we have seen in simple linear regression, the overall quality of the model can be assessed by examining the R-squared (R2) and Residual.

*R-squared*:

This measure is defined by the proportion of the total variability explained by the regression model. This metric can seem a little bit complicated, but in general, for models that fit the data well, R² is near 1. Models that poorly fit the data have R² near 0. In the examples below, the first one has an R² 0.3688, this means that the model explains only 36% of the data variability. A problem with the R2, is that, it will always increase when more variables are added to the model, even if those variables are only weakly associated with the response. As you have seen in the simple linear regression, a solution is to adjust the R2 by taking into account the number of predictor variables.

The adjustment in the “Adjusted R Square” value in the summary output is a correction for the number of x variables included in the prediction model. We can conclude that this model does perform better than the simple linear model with only hits, which had an adjusted R2 of 0.0382.

*Residual* You can have a pretty good R² in your model, but let’s not rush to conclusions here. Let’s see in our model how they look like. Ideally, when you plot the residuals, they should look random. Otherwise means that maybe there is a hidden pattern that the linear model is not considering. To plot the residuals, use the command plot(lmTemp$residuals).

plot(baseball\_multiple,1)

A close up of text on a white background

Description automatically generated

### Exercise 5 – Now it is your turn: You are asked to examine which linear equation is better at predicting a player’s salary:

1. **Salary = B0+B1RBI+ B2CAtBat + B3\*Errors**
2. **Salary = B0+B1Walks+ B2CAtBat+ B3\*Errors**

Conduct the appropriate analysis to answer the question by completing the following steps:

- Visualise the relationships in two separate scatterplots

- Conduct the two separate appropriate analyse(s) using lm()

- Print the test statistic for each coefficient

- Compare the outputs of these two models. What do you find?

## The last part of our Workshop Series ☺

In this last part of workshop 5 you will get a basic introduction to logistic regression. Note that this will be less extensive than the intro into simple and multiple regression. It will cover:

* The essence of logit models
* Simple Logistic regression: Predicting the probability of response Y with a single explanatory variable X
* Multiple Logistic regression: Predicting the probability of response Y with multiple predictor variables

## What is Logistic Regression?

Logistic regression (aka logit regression or logit model) is a regression model where the dependent variable Y is NOT continuous (as is the case with simple and multiple regression), but categorical. Logistic regression allows us to estimate the probability of a categorical response based on one or more explanatory variables (X). It allows us to say that the presence of a specific explanatory variable predicts an increase (or decreases) in the probability of a given outcome (the dependent variable) by a specific percentage. This tutorial covers the case when Y is binary — that is, where it can take only two values, “0” and “1”, which represents outcomes such as pass/fail, win/lose, males/females or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed with multinomial logistic regression (note this is different to multiple logistic regression(!), or, if the multiple categories are ordered, in ordinal logistic regression. However, this is out of the scope of this introduction.

### Case Study

A Data Science Researcher in the University of Maastricht is interested in investigating how variables, such as statistics exam score, responsible exam score, prestige of the undergraduate institution and gender, impact on admission rate into the new graduate program launched at the Institute of Data Science called Responsible Data Science. We are going to work with a dependent binary variable (y) that has two values: admit/don’t admit.

#### Description of the data

We have generated hypothetical data, which can be obtained from this link: <https://docs.google.com/spreadsheets/d/1QgksUXsOq2Lz7UyHAvdGTZozUDZd5mUuKM3WPn5q_Go/edit#gid=0>

This dataset has a binary response (outcome/ dependent) variable called admit. There are four explanatory variables: stats, rank, responsible female. We will treat the variables stats and responsible as continuous. The variable `rank` takes on the values 1 through 4. Institutions with a rank of 1 have the highest prestige, while those with a rank of 4 have the lowest. The female variable is 0 when male and 1 in case is female.

*Import the data*:

library(readxl)  
ResponsibleDataScience <- read\_excel("InstituteofDataScience.xlsx")

Check the ResponsibleDataSciencevariables:

str(ResponsibleDataScience)

## tibble [400 x 5] (S3: tbl\_df/tbl/data.frame)  
## $ admit : num [1:400] 0 1 1 1 0 1 1 0 1 0 ...  
## $ responsible: num [1:400] 380 660 800 640 520 760 560 400 540 700 ...  
## $ stats : num [1:400] 3.61 4.67 5 4.19 2.93 4 3.98 3.08 4.39 3.92 ...  
## $ rank : num [1:400] 3 3 1 4 4 2 1 2 3 2 ...  
## $ female : num [1:400] 0 1 1 1 0 1 1 0 1 0 ...

We can get basic descriptives for the entire data set by using summary()

summary(ResponsibleDataScience)

## admit responsible stats rank   
## Min. :0.0000 Min. :220.0 Min. :2.260 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:520.0 1st Qu.:3.217 1st Qu.:2.000   
## Median :0.0000 Median :580.0 Median :3.585 Median :2.000   
## Mean :0.3175 Mean :587.7 Mean :3.707 Mean :2.485   
## 3rd Qu.:1.0000 3rd Qu.:660.0 3rd Qu.:4.173 3rd Qu.:3.000   
## Max. :1.0000 Max. :800.0 Max. :5.000 Max. :4.000   
## female   
## Min. :0.00   
## 1st Qu.:0.00   
## Median :0.00   
## Mean :0.33   
## 3rd Qu.:1.00   
## Max. :1.00

#### 

#### The essence of logit models

Let’s first have a look on the relationship between the admit (y) and stats (x) variable. Whether or not stats is related to the student being admitted in the graduate program:

## Plotting the data  
library(ggplot2)  
ggplot(ResponsibleDataScience, aes(stats, admit)) +  
 geom\_point()

A close up of a white wall

Description automatically generated

So each dots represent different students and how its grade in stats is connected with whether or not they were admitted into the Graduate Program. Value 1 are students that were admitted and value 0 not admitted.

In order to fully understand the differences between linear and binomial regression, we can model this relationship using a linear regression model for example. Let’s try it and add it on the plot:

ggplot(ResponsibleDataScience, aes(stats, admit)) +  
 geom\_point() +  
 geom\_smooth(method = 'lm', se = FALSE) + # linear regression and not plot the standard error  
 coord\_cartesian(ylim = c(0,1)) # limit the plot

## `geom\_smooth()` using formula 'y ~ x'

A close up of a person

Description automatically generated

Super, quite an interesting plot! The linear model is not really a good fit right? Indeed, it is  violating some of the assumptions for linear model such as linearity (i.e. this plots shows non linear relationship between the variables) and  homoscedasticity (we could expect some that our residuals varies across values of our independent variable). You are always required to fully check these assumptions out before conducting any statistical analysis. HOWEVER, for the sake of simplicity, today we are not going to deal with such exercise.

Want to know how to deal with that? To get more information about it please this short tutorial: [http://r-statistics.co/Assumptions-of-Linear-Regression.html](https://slack-redir.net/link?url=http%3A%2F%2Fr-statistics.co%2FAssumptions-of-Linear-Regression.html" \t "_blank)

Thus, ignoring the fact that assumptions have been violated, lets convert the blue line into a binomial graph/curve in the same plot but now, with the method glm(Generalized Linear model as a family models that are not linear, such logistic model):

ggplot(ResponsibleDataScience, aes(stats, admit)) +  
 geom\_point() +  
 geom\_smooth(method = 'glm', se = FALSE, method.args = list(family = 'binomial')) + # and not plot the standard error  
 coord\_cartesian(ylim = c(0,1)) # limit the plot

## `geom\_smooth()` using formula 'y ~ x'

A picture containing man, white

Description automatically generated

*Note*: if you get this `geom\_smooth() using formula ‘y ~ x’`, dont worry much, your plot should be correctly displayed in your R studio.

As you might notice, this blue line appears to fit our data much better. Instead of linear regression(e.g. straight line) we just incorporated a binomial `curve`. So this is essentially what we do in this modelling; this model looks a little bit better when predicting admit rate based on stats scores. Indeed, the binomial model will predict those values closer to 0 as not addmitted, whereas it will predict students as admitted if the value is around 1.

#### Logistic regression with a single continuous predictor variables

Let’s build a simple logistic regression formally, and we can see how likely a student is admitted in the graduate program with a one unit change in stats score. Let’s create a logitmodel to save our results as follow. The type of code is pretty similar to the linear regression one (though not identical ;)):

# MODEL ADMIT BY STATS  
logitmodel <- glm(formula = admit ~ stats, # response variable ~ explanatory variable  
 data = ResponsibleDataScience,  
 family = 'binomial')

Go ahead and look at the models output:

# call the result object  
summary(logitmodel)

##   
## Call:  
## glm(formula = admit ~ stats, family = "binomial", data = ResponsibleDataScience)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.1613 -0.1277 -0.0270 0.0424 3.1920   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -34.869 4.771 -7.308 2.71e-13 \*\*\*  
## stats 8.708 1.205 7.227 4.93e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 499.98 on 399 degrees of freedom  
## Residual deviance: 117.27 on 398 degrees of freedom  
## AIC: 121.27  
##   
## Number of Fisher Scoring iterations: 8

### Model Output Interpretation

* Call, this is R reminding us what the model we ran were the options we specified.
* Deviance residuals are a measure of model fit. This part of our output shows the distribution of the deviance residuals for individual cases used in the model.
* Coefficients with their standard errors, the z-statistic (sometimes called a Wald z-statistic), and the associated p-values. Both intercept and stats are statistically significant. We can see the coefficient models from the model output:

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -34.869 4.771 -7.308 2.71e-13

stats 8.708 1.205 7.227 4.93e-13

The 8.70764 is the log odds of the variable stats and means that for a one unit increase in stats (in this case one grade), the log odds of being admitted to graduate school increases by 0.804.

To derive the exact odds ratios from log odds you need to take the exponent of the coefficient (to do the opposite of log):

exp(coef(logitmodel))

## (Intercept) stats   
## 7.190987e-16 6.049001e+03

Informally speaking, this number here is telling us that for one unit increase in stats (thus one grade), a student is 6049 times more likely to be admitted in the graduate program. So that is the way we interpret odd ratios. Learn more about log odds and odds ratios here:

* <https://stats.idre.ucla.edu/other/mult-pkg/faq/general/faq-how-do-i-interpret-odds-ratios-in-logistic-regression/>

**Exercise 6 – Now it is your turn: Using the previous explained steps, run a simple logistic regression model using *responsible* score as the independent variable and *admit* as the dependent variable. Interpret the results.**

***Using the previous steps, replicate a simple logistic regression model by adding responsible score variable into the model (use the below R code template)*.**

#### 

## Multiple Logistic regression

When dealing with logistic regression models, it can be possible that your explanatory variables are continuous but they can also be categorical. So, the next step is to create a model that predicts admissions based on both, categorical and continous variables.

At this time, we want to predict admit based on stats and rank. Let’s call it logitmodel1. It is our second logistic model. But first, remember to convert into factor the variable rank as following:

# convert into a factor  
ResponsibleDataScience$rank <- factor(ResponsibleDataScience$rank)

Now, we need to type a R command that stores the model results. First create a object called logitmodel1 and then summary() :

# create the second model  
logitmodel1 <- glm(formula = admit ~ stats + rank, # binomial formula with the two explanatory variables  
 data = ResponsibleDataScience,  
 family = 'binomial')

Again, we can get our results with summary():

summary(logitmodel1)

##   
## Call:  
## glm(formula = admit ~ stats + rank, family = "binomial", data = ResponsibleDataScience)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.43505 -0.12115 -0.02895 0.04091 2.97136   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -33.8273 4.7847 -7.070 1.55e-12 \*\*\*  
## stats 8.6038 1.2010 7.164 7.85e-13 \*\*\*  
## rank2 -0.1914 0.7184 -0.266 0.7899   
## rank3 -1.3280 0.7618 -1.743 0.0813 .   
## rank4 -1.0323 0.9072 -1.138 0.2551   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 499.98 on 399 degrees of freedom  
## Residual deviance: 111.87 on 395 degrees of freedom  
## AIC: 121.87  
##   
## Number of Fisher Scoring iterations: 8

### Model Output Interpretation

As in the other outputs discussed so far, we have coefficients for each of our explanatory variables and the correspondent p-values. You can easily identify that stats is highly significant since the p-value is close to zero. That means that we are certainly sure that its estimated coefficient is 8.6063.

But what is interesting for us now is the rank variable. We have three coefficients and the reason for this is that rank is a factor with four levels so these coefficients will be compared always against the *reference or baseline level*. This happens because we have defined rank as a factor before running this analysis. We had to because without having done this, we would have just added the variable ‘rank’ as continuous which is conceptually wrong. The way that we would interpret this, (the coefficient of rank variable) is for example: having attended an undergraduate institution with rank 2, versus an institution with rank 1, changes the log odds (chances) of admission by -0.1914. The log odds of admission decreases by 1.32 as we go from rank 1 to rank 3. Now practically, all of this means that the higher the prestige of the undergraduate university, the more likely it is that a student is admitted. Although it should noticed, that none of them are statistically significant for our model (!).

As before, we are going to convert our logs odds into odds ratios to achieve a better interpretation of what we are saying:

exp(coef(logitmodel1)) # calculate odds ratios

## (Intercept) stats rank2 rank3 rank4   
## 2.036993e-15 5.452324e+03 8.257842e-01 2.649940e-01 3.561867e-01

So, for one unit increase in stats score, the student is 5.452 times more likely to be admitted in the graduate program. The rest of odds ratios that correspond to rank variables are quite difficult to interpret since they are smaller than 1. Often, the interpretation is done by calculating the inverse of the number that are smaller than 1. For instance, rank2 is 8.257842e-01. We calculate the inverse by:

1/8.257842e-01

## [1] 1.21097

For rank2 then, a student was 1.33 times LESS likely to be admitted than a student with an undergraduate in the rank 1. As the odds ratios for level rank 3 and rank 4 are also below 1, we calculate the inverse for these as well.

#rank3  
1/2.649940e-01

## [1] 3.77367

A student was 3.77 times less likely to be admitted if she took her undergraduate courses in rank 3 than rank 1 university.

#rank4  
1/3.561867e-01

## [1] 2.807516

A student was 2.80 times less likely to be admitted if she took her undergraduate courses in rank 4 than rank 1 university.

**Exercise 7 – Now it is your turn: Using the previous explained steps, run a multiple logistic regression model using the following variables as independent variables: *responsible* scores, *stats* scores and *female.* Use *admit* as the dependent variable. Interpret the results.**

## Conclusion

You made it to the end! Logistic regression is a big topic, and it's here to stay. Here I presented a few tricks that can help to tune and take the most advantage of such powerful algorithm, yet so simple. You also learned how to understand what's behind this simple statistical model and how you can modify it according to your needs. You can also explore other options by typing “?glm” on the R console and looking at the different parameters not covered in here. Hope you enjoyed it!

## Appendix

#### Variable description

| **Name** | **Description** |
| --- | --- |
| AtBat | Number of times at bat in 1986 |
| Hits | Number of hits in 1986 |
| HmRun | Number of home runs in 1986 |
| Runs | Number of runs in 1986 |
| RBI | Number of runs batted in in 1986 |
| Walks | Number of walks in 1986 |
| Years | Number of years in the major leagues |
| CAtBat | Number of times at bat during his career |
| CHits | Number of hits during his career |
| CHmRun | Number of home runs during his career |
| CRuns | Number of runs during his career |
| CRBI | Number of runs batted in during his career |
| CWalks | Number of walks during his career |
| League | A factor with levels A and N indicating player’s league at the end of 1986 |
| Division | A factor with levels E and W indicating player’s division at the end of 1986 |
| PutOuts | Number of put outs in 1986 |
| Assists | Number of assists in 1986 |
| Errors | Number of errors in 1986 |
| Salary | 1987 annual salary on opening day in thousands of dollars |
| NewLeague | A factor with levels A and N indicating player’s league at the beginning of 1987 |