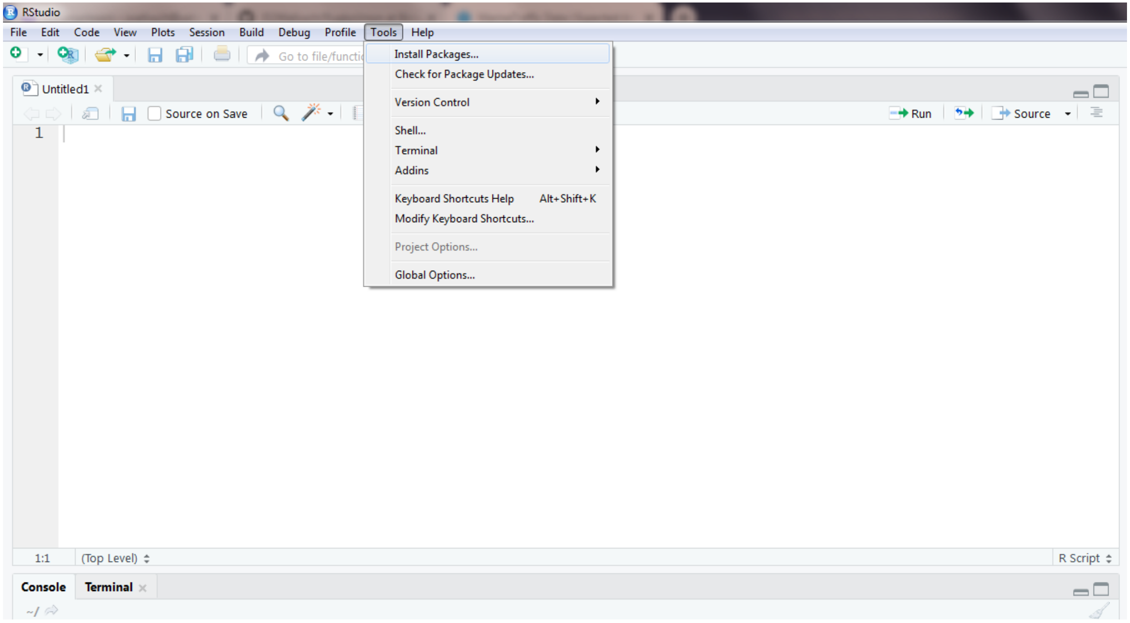
Binary-Choice Model

Step 0: Download R and RStudio

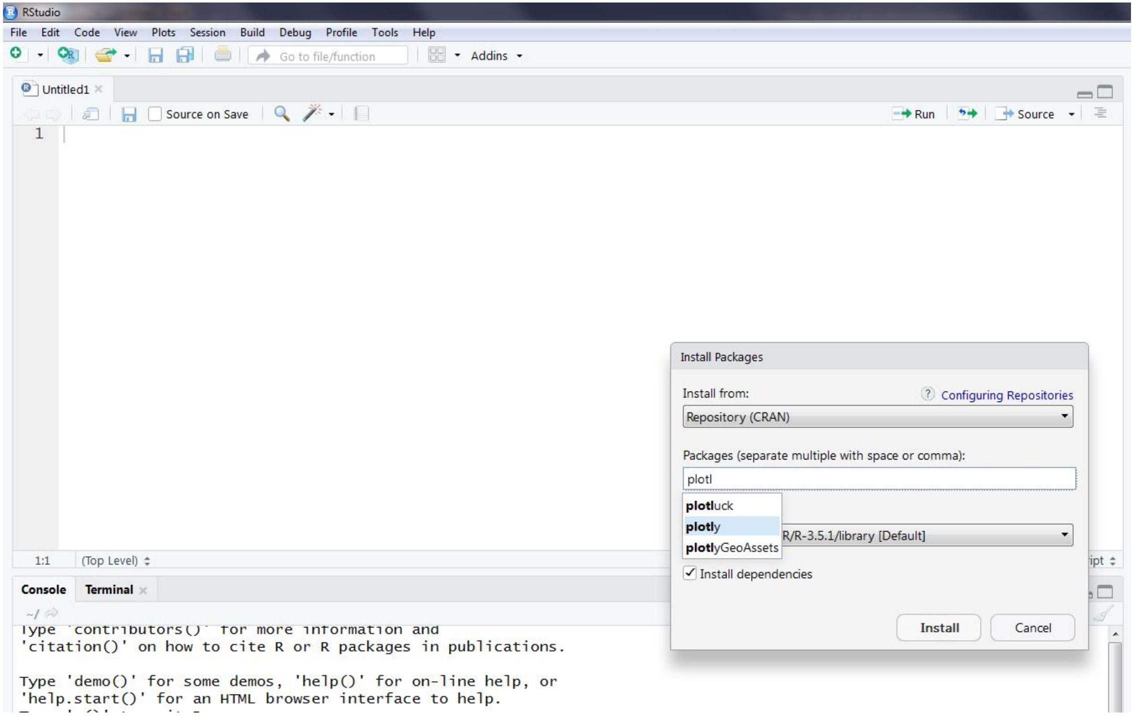
R: https://cran.r-project.org/
RStudio: https://rstudio.com/

Install R packages

1. Open RStudio
2. Click Tools/Install Packages



3. Start typing **plotly** (that’s the first package you’re going to install) in the Packages window. Once plotly installation is finished, repeat it for **moments** package.



Install from: Repository (CRAN)

Packages (separate multiple with space or comma): plotl

R/R-3.5.1/library [Default]

☑ Install dependencies

```
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
```

4. Copy the R script from Github and paste in the RStudio (upper left box)

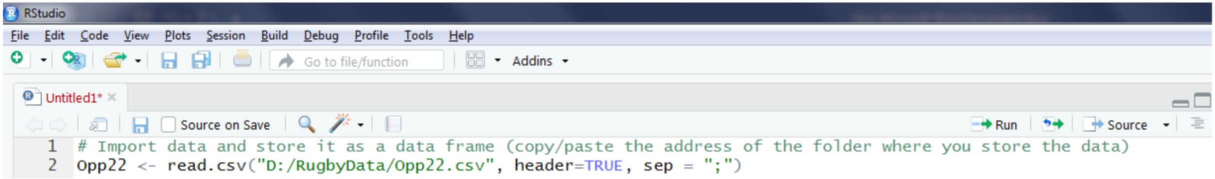
Step 1: Binary Choice Models

1. The explaining variable in a binary-choice model is expressed as a 0-1 dummy. It means that the modelled events are binary and mutually exclusive.
2. Examples: Rugby
    a. Line-out
        i. line-out won = 1
        ii. line-out stolen by the defence = 0
    b. Scoring points after entering the Opp22
        i. scoring = 1
        ii. not scoring = 0
    c. A multi-pass play off the attacking ruck
        i. two or more passes (incl. the ground pass) = 1
        ii. carry after the ground pass = 0
3. Example: Football
    a. Play-calling
        i. run = 1
        ii. pass = 0
    b. Probability that the defence blitzes in the 3rd down
        i. blitzing = 1
        ii. not blitzing = 0
4. Make sure the event you want to model is set to 1. For instance, if you want to model the probability of winning an attacking line-out, don't set the line-out stolen to 1. Of course it's still possible to re-calculate the probability of winning even though they were set to 0, but it would require additional calculations.
5. Make sure the event you want to model has occurred fairly frequently, otherwise the result will be biased
    a. Example: probability of stealing a line-out, Harlequins (PremRugby 2019/20, rounds 1-5). Line-outs stolen by the Quins – 4 (out of 53). The model will struggle to predict any stolen line-outs.
6. Any metric that can be expressed numerically (also as a 0-1 dummy) can serve as an explanatory variable.
    a. Example: Rugby
        i. Probability of scoring a try after entering the Opp22:
            1. number of phases
            2. number of passes (incl. ground passes)
            3. number of pick&goes (all variables are expressed as integers and thus can be implemented directly without any manipulation)
    b. Example: Football
        i. 3rd down blitzing: what is the probability that the defensive co-ordinator will call a blitz when
            1. distance: 6 yds,
            2. safeties: single-high
            3. offensive personnel: empty (distance is a continuous variable and thus can be implemented directly, safeties and personnel need to be turned into dummies)

    c. Basic requirements
        i. make sure there's (much) more observations than explanatory variables
        ii. try avoiding 0-1 dummy variables

Step 2: Run the Script

1. Download the csv file
2. Say the folder you store the file is in the *RugbyData* folder on the hard drive D
3. Paste the script to the RStudio console (upper-left quadrant). Type the address of the folder, in which you store the data, in line 2:



```
1  # Import data and store it as a data frame (copy/paste the address of the folder where you store the data)
2  Opp22 <- read.csv("D:/RugbyData/Opp22.csv", header=TRUE, sep = ";")
```

4. Set the cursor at the end of the line you want to run. (For multi-line functions, either highlight the function or set the cursor at the end of the function). Click the Run-button or use the key combination Ctrl-Enter

Step 3: Filtering

I filtered out all entries into the Opp22 without the controlled possession of the ball. Example: line 17 in the csv file – Portugal had conceded a penalty that led to kicking for touch and a line-out, but the line-out throw resulted in a knock-on. The expression in the line 9 of the script helps to filter out all entries flagged as N.
Notice that the variables (stored in columns of the csv file) are called by using the $-sign. The data is stored in a data frame Opp22 (see line 2 of the script; a data frame can store multiple data tables of the same length), and the columns are named exactly as in the cvs-file.

Step 4: Descriptive Statistics

Descriptive stats are calculated in lines 12-16. Highlight those lines and click the Run-button (or use the key combination Ctrl-Enter). Then highlight the metric and click Run again. For instance, if you want to know what the median phases in the Opp22 was, highlight `med22` (line 13) and click Run. Notice how I calculated skewness in line 16. As the function skewness is used in multiple package, I use `moments::skewness()` to make sure the moment-package is employed.

Step 5: Create the Binary Dummy

In order to create the dummy variable I used the `ifelse()` function. There are three arguments you need to specify for this function:

1. Test – what do you want to test. In this case we test if the attacking team scored points after entering the Opp22 by setting `Opp22_Y$pts > 0`
2. Yes – value that will appear if the test outcome is TRUE
3. No – value that will appear if the test outcome is FALSE

Step 6: Estimations

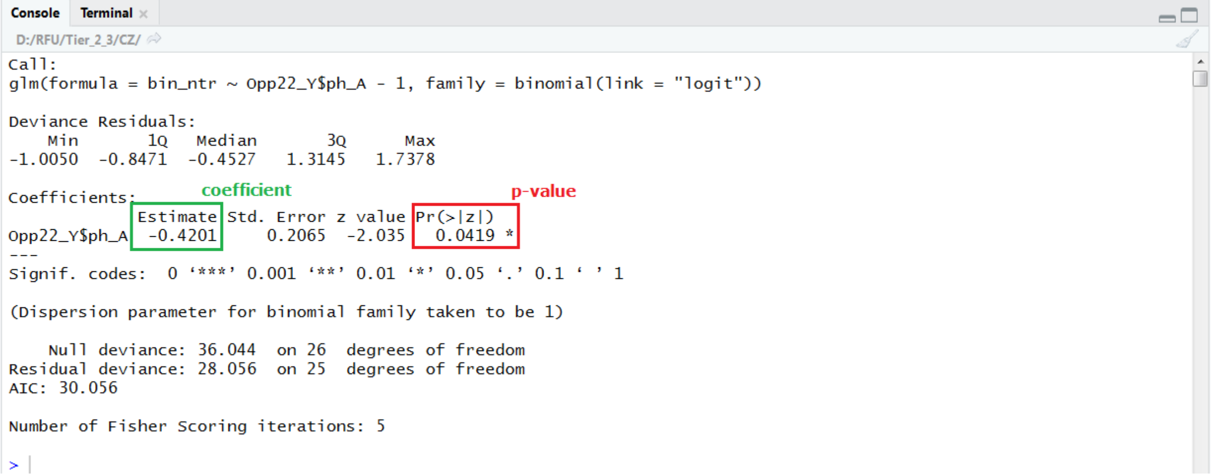
The model is specified in lines 25-27. I regressed the dummy variable `bin_ntr` against the number of phases after entering the Opp22 (`Opp22_Y$ph_A`). I removed the constant term from the set of explanatory variables (notice the expression: – 1 in line 26). Significance of the estimated coefficient is tested in lines 31-35 (standard routine is to check the significance at 1%, 5%, and 10% significance level, but you should never trust this standard routine; in the nearest future I will elaborate more on that topic).

Step 7: Simulating Probabilities

In the final step, I simulate the probabilities. First, I extract the estimated coefficient. Then, I generate a sequence of integers serving as phases after entering the Opp22. Notice that the sequence starts with 1 and ends up with maximum number of phases + 1. Finally, I use logistic probability distribution to simulate the series of probabilities for specific number of phases. The simulations are presented graphically (lines 43-49).

Step 8: Interpretation

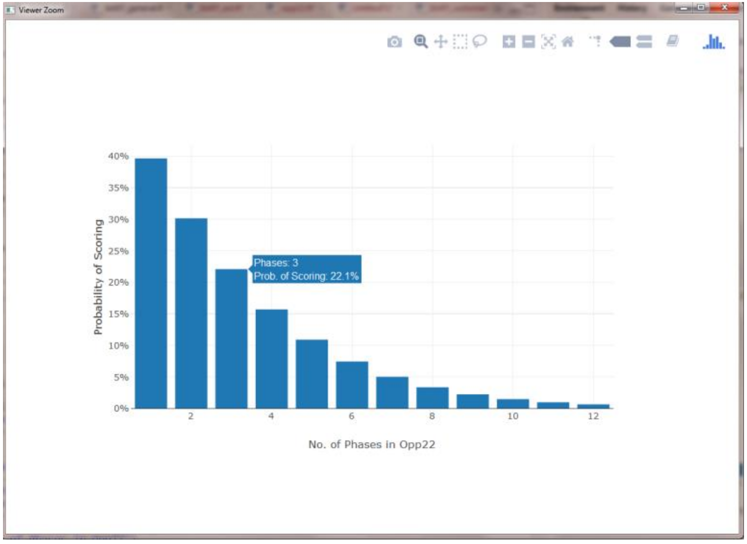
I always start the interpretion with the tests for significance. Testing for coefficient's significance is actually a test for insignificance, because the hypothesis reads *the coefficient is equal to 0* (hence, if it's equal to zero its impact on the explaining variable is 0, so it's not significant). Let's go back to line 29. Run it and check the RStudio console (bottom right quadrant), where the estimation results are printed.



```
Call:
glm(formula = bin_ntr ~ Opp22_Y$ph_A - 1, family = binomial(link = "logit"))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0050  -0.8471  -0.4527   1.3145   1.7378

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
Opp22_Y$ph_A  -0.4201     0.2065  -2.035   0.0419 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 36.044  on 26  degrees of freedom
Residual deviance: 28.056  on 25  degrees of freedom
AIC: 30.056

Number of Fisher Scoring iterations: 5

>
```

The p-value (aka empirical probability) for the significance test is marked red. Think of it as the probability that the coefficient is equal to 0. The p-value is small (0.042), hence there's a highly significant relationship between number of phases and probability of scoring points. Run line 31 to extract the p-value. After running lines 32-35 additional information will be printed in the console.

In the next step inspect the coefficient's sign (marked green above). The sign is negative, so the more phases is played the smaller the probability of scoring. Now bear in mind that the model is not linear, the coefficient has no direct interpretation, so you need an intermediary step to calculate the probability. (That's why we needed line 40, where the logistic probability distribution was implemented). The probabilities of scoring depending on the number of phases played in the Opp22 are estimated. A word of advice: when it come to presenting numbers, always do it graphically.



The plotly figure depicts the simulated probabilities of scoring points (vertical axis) for number of phases in the Opp22. The graph will appear in the RStudio viewer (bottom right quadrant), but you can also magnify it (click zoom to view the graph in a separate box) or save it as jpeg/html). The graph is interactive. Hover over its surface to display the phases and probabilities. One glimpse at the plotly figure and we see that the probability of scoring decreases in a non-linear manner as the attacking team plays more phases. For the defence, it’s crucial to survive 4 phases, as the probability of losing points drops below 0.2.