MVA: An outbreak of gastroenteritis in Stegen, Germany. Logistic regression in R

Patrick Keating (AGES), Niklas Willrich (RKI) and Alexander Spina (AGES)

28 February 2017

**Contributors to *R* code:**  
Daniel Gardiner (PHE) Lukas Richter (AGES)

The following code has been adapted to *R* for learning purposes. The initial contributors are listed below. All copyrights and licenses of the original document apply here as well.

**Authors:**  
Alain Moren and Gilles Desve

**Reviewers:**  
Marta Valenciano, Alain Moren.

**Adapted for the EPIET MVA module December**

* **2015:** Alicia Barrasa (EPIET), Ioannis Karagiannis (UK-FETP)
* **2016:** Alicia Barrasa (EPIET), Ioannis Karagiannis (PHE) and Thomas Inns (PHE): The use of the glm command and the mathematical representation of the models have been added
* **2017:** Alicia Barrasa (EPIET) and Thomas Inns (PHE): questions were rephrased to reflect real life scenarios (rather than academic exercise)

# Prerequisites

Participants are expected to be familiar with data management as well as descriptive and stratified analysis in R

# An introduction to the R companion

This text was adapted from the introduction used at the 2016 TSA module.

R packages are bundles of functions which extend the capability of R. Thousands of add-on packages are available in the main online repository (known as CRAN) and many more packages in development can be found on GitHub. They may be installed and updated over the Internet.

We will mainly use packages which come ready installed with R (base code), but where it makes things easier we will use add-on packages. In addition, we have included a few extra functions to simplify the code required. All the R packages you need for this case study can be installed over the Internet.

# Installing required packages for this case study  
required\_packages <- c("foreign", "epiR", "multcomp", "broom")   
install.packages(required\_packages)

Run the following code at the beginning of the case study to make sure that you have made available all the packages that you need. Be sure to include it in any scripts too.

# Loading required packages for this case study  
required\_packages <- c("foreign", "epiR", "multcomp", "broom")   
  
for (i in seq(along = required\_packages))  
 library(required\_packages[i], character.only = TRUE)

R and Stata have minor differences in default settings and methods. In this document we will follow the Stata analysis as closely as possible, but small and usually unimportant differences may be noted between the statistical findings in R and those in Stata (e.g. in the 95% confidence intervals obtained in regression models). At some points additional steps (which would usually be optional in R) will be taken to produce output which is comparable to that of Stata.

You will work with Stata.dta data sets which can be loaded into R with the "foreign" or "readstata13" packages. The appropriate functions to use will be indicated.

R can hold one or many data sets in memory simultaneously, so there is usually no need to save intermediate files or close and re-open datasets.

# 

# Session 1 - Logistic regression: adjusting for confounding

## Question 8. How would you explore the effect of several risk factors? How would you account for dose response?

## Help Q8

The objective of your multivariable analysis is to identify variables independently associated with the outcome and to control for confounding.

To prepare your dataset for multivariable analysis, you need to decide on the variables of interest based on your prevoious descriptive and stratified analysis and you might need to create or recode variables (age groups, dummy variables, etc...).

Start a new R script, name it **logistic.r** and save it in your working directory. Write all commands in the R script so that you can run (and re-run) it when needed during the exercise.

Open the **tiraclean.dta** dataset

tira.data <- read.dta("tiraclean.dta", convert.factors = FALSE)

#### Logistic regression using tiramisu as a dichotomous variable:

Using the **generalised linear model** (glm) function with the **logit link** will provide similar output to the logistic command in Stata.

# For regression analysis in R, it is important that your variables are ordered 0,1 - i.e. not as (1,0).  
# Confirming the order of the tira variable  
table(tira.data$tira)

##   
## 0 1   
## 165 121

# Create logit regression model with tira as exposure variable  
  
model1 <- glm(ill~tira,  
 data = tira.data,  
 family = binomial(link = "logit"))  
  
# Gives an overview of key elements of the model  
summary(model1)

## Call:  
## glm(formula = ill ~ tira, family = binomial(link = "logit"),   
## data = tira.data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7320 -0.2944 -0.2944 0.7106 2.5140   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.1167 0.3862 -8.071 7e-16 \*\*\*  
## tira 4.3641 0.4436 9.837 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 371.44 on 285 degrees of freedom  
## Residual deviance: 186.41 on 284 degrees of freedom  
## (5 observations deleted due to missingness)  
## AIC: 190.41  
##   
## Number of Fisher Scoring iterations: 5

You can write down the above model by substituting α and β with the coefficients above.

ln(p/1-p)) is the log of the odds for the outcome

* α is the log of the odds in the unexposed
* β is the log of the OR for exposure x

log odds = -3.11+(4.36 \* tira)

To obtain the ORs of your coefficients, you will need to do the following:

* save your model output as an object
* use the tidy function of the broom package
* exponentiate the output

# Obtaining the key output of the regression model including ORs and CIs  
model1op <- tidy(model1, exponentiate = TRUE, conf.int = TRUE)  
model1op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.0443038 0.3861778 -8.070596 6.995604e-16 0.01880027  
## 2 tira 78.5820052 0.4436312 9.837322 7.775296e-23 35.02913548  
## conf.high  
## 1 0.08744858  
## 2 203.48219359

The output of model1op is similar to what you would obtain using the **logistic** command in Stata. The estimates are the same as Stata, although there will be small differences in the 95% CI.

That model corresponds to the equation  
odds = exp(α + βX) =

The (Intercept) is exp(α), which in cohort studies can be interpreted as the odds of being a case among the unexposed; in case control studies the interpretation is meaningless. However, the (Intercept) is not an OR. This odds needs to be multiplied with the correct odds ratios for each exposure group to produce the odds of being a case for each exposure combination.

The OR=78.58 corresponds to exp(β) in the equation above.

#### Logistic regression using tportion as a categorical variable

As seen below, there are **4 levels** to tportion: 0 to 3 portions.

# The levels of tportion  
table(tira.data$tportion)

##   
## 0 1 2 3   
## 165 65 42 14

R will automatically take the lowest value of tportion as the reference category.

# Here we specify that tportion is a factor variable  
model2 <- glm(ill~factor(tportion),   
 data = tira.data,  
 family = binomial(link = "logit"))  
  
model2op <- tidy(model2, exponentiate = TRUE, conf.int = TRUE)  
  
# The rownames of the output say "factor" because the glm model believes the var name is "factor(tportion)"  
model2op

## term estimate std.error statistic p.value  
## 1 (Intercept) 0.0443038 0.3861778 -8.070596 6.995604e-16  
## 2 factor(tportion)1 47.2925137 0.4684865 8.231511 1.848660e-16  
## 3 factor(tportion)2 214.4285567 0.6522646 8.229753 1.875991e-16  
## 4 factor(tportion)3 135.4285621 0.8558426 5.735218 9.738688e-09

## conf.low conf.high  
## 1 0.01880027 0.08744858  
## 2 19.96920519 127.75259273  
## 3 66.51598007 890.06235491  
## 4 30.21489385 992.29211228

We can however change the reference level (for example use 3 portions instead of 0).  
*NB*: reference designates the index and not the value.

# Make a new variable tportion2, where we select the 4th level as the reference  
tira.data$tportion2 <- relevel(factor(tira.data$tportion), ref = 4)  
  
# Run the logit model with this new variable  
model3 <- glm(ill ~ tportion2,   
 data = tira.data,  
 family = binomial(link = "logit"))  
  
model3op <- tidy(model3, exponentiate = TRUE, conf.int = TRUE)  
model3op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 6.000000000 0.7637626 2.3459638 1.897795e-02 1.636237363  
## 2 tportion20 0.007383967 0.8558426 -5.7352182 9.738688e-09 0.001007768  
## 3 tportion21 0.349206349 0.8085046 -1.3012818 1.931620e-01 0.051342118  
## 4 tportion22 1.583333333 0.9271726 0.4956276 6.201572e-01 0.201642583  
## conf.high  
## 1 38.56005530  
## 2 0.03309626  
## 3 1.43404252  
## 4 9.22304424

What would have happened if we had included tportion without indicating that it is categorical? Try it and interpret the OR.

# Regression using tportion as continuous variable  
model4 <- glm(ill ~ tportion,  
 data = tira.data,  
 family = binomial(link = "logit"))  
  
model4op <- tidy(model4, exponentiate = TRUE, conf.int = TRUE)  
model4op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.08188365 0.2761290 -9.062632 1.273396e-19 0.0456652  
## 2 tportion 14.21641159 0.2983238 8.897705 5.701355e-19 8.2516286  
## conf.high  
## 1 0.1357736  
## 2 26.6992874

Remember that the logistic equation can be expressed as:  
odds =

The coefficient 14.21 represents the increase in the OR with one unit increase in tportion. What would be the OR for a two-unit increase in tportion?

#### Adding a second variable to the model

# We add beer to the model  
model5 <- glm(ill ~ tira + beer,  
 data = tira.data,  
 family = binomial(link = "logit"))  
  
model5op <- tidy(model5, exponentiate = TRUE, conf.int = TRUE)  
model5op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.0633450 0.4025692 -6.853876 7.187562e-12 0.02615275  
## 2 tira 74.0274332 0.4547373 9.465763 2.914251e-21 32.32622013  
## 3 beer 0.4689017 0.4028665 -1.879933 6.011715e-02 0.20957970  
## conf.high  
## 1 0.129585  
## 2 195.934241  
## 3 1.026347

Odds[illness] =

Note that, in the above expression, tira and beer can have the values 0 or 1, according to whether they consumed tira or beer respectively.

(Intercept) = 0.063 is the odds of illness among the unexposed, i.e. among those who consumed neither tiramisu nor beer.

exp(β1) = 74.02 is the OR for tiramisu adjusted by beer. The odds of illness among those who consumed tiramisu but did not drink beer is 74 times higher compared to those who consumed neither tiramisu nor beer.

exp(β2) = 0.47 is the OR for beer adjusted by tiramisu. The odds of illness among those who drank beer but did not consume tiramisu is almost half the odds of those who consumed neither tiramisu nor beer; however, this finding is not statistically significant.

The odds of illness among those who ate tiramisu and beer is 74.02\*0.47 times higher than among those who consumed neither.

#### Adding a third variable to the model

We add mousse to the previous model. To simplify matters, we can use the **update** function. In this way, we retain the dataset and family from the previous model and just have to specify the variables to include in the formula.

model6 <- update(model5,  
 formula = ill ~ tira + beer + mousse)  
  
model6op <- tidy(model6, exponentiate = TRUE, conf.int = TRUE)  
model6op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.0512656 0.4285911 -6.931398 4.167012e-12 0.02016546  
## 2 tira 47.7558996 0.4879683 7.922856 2.321159e-15 19.51012482  
## 3 beer 0.5129572 0.4092824 -1.631057 1.028783e-01 0.22679653  
## 4 mousse 2.3395140 0.4275537 1.987922 4.682037e-02 0.99001885  
## conf.high  
## 1 0.1103337  
## 2 134.6657355  
## 3 1.1393003  
## 4 5.3547326

Try to write down the model and interpret all its coefficients.

#### Adding variables in a step-by-step fashion using the anova and chi2 test to compare different models

Variables to be included in a multivariable regression model are selected on the basis of the results of the crude analysis. Variables showing an association with the outcome and having a p-value less than 0.2 are often considered eligible. The cut-off should be chosen depending on the specific situation. Often it is between 0.25 and 0.1 but higher p-values can sometimes be justified. However, if you have any reason to believe a specific variable (exposure) should be in the model (i.e. because it might be a confounder), you should include it in the model anyway. There is no golden rule in the final inclusion of variables in a multivariable analysis model, especially in outbreak investigations.

To be able to statistically check if the inclusion of a variable improves the model significantly, the models need to have the same number of observations. If you remember for some variables we had missings, meaning that each of them have a different number of observations. You need to drop all the missings.

We can do this by looping over the variables and dropping rows with NAs.

# Drop observations with missing data   
# List all the variables for which the missing values will be dropped  
vars <- c("ill", "tira", "age", "dmousse", "wmousse", "beer", "fruitsalad", "redjelly", "tportion", "mportion", "salmon", "mince", "tomato", "horseradish", "chickenwin", "roastbeef", "pork")  
  
# Create a new data set and assign the original data set to it!  
tira.data.new <- tira.data  
  
#Only keep the values of tira.data.new that are NOT equal to NA  
for (var in vars) {  
 tira.data.new <- tira.data.new[!is.na(tira.data.new[,var]),]   
}  
  
# You should have 239 observations after completion of this loop

There are two possible strategies:

* to start off with a model that includes only one independent variable and add others one by one
* to start with a full model (including all eligible variables) and, one at a time, remove variables that do not seem relevant.

We will begin with only one independent variable.

# Only one independent variable  
model7 <- glm(ill~tira,  
 data = tira.data.new,  
 family = binomial(link = "logit"))  
  
model7op <- tidy(model7, exponentiate = TRUE, conf.int = TRUE)  
model7op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.05185185 0.3876173 -7.634759 2.262439e-14 0.02195746  
## 2 tira 74.24999945 0.4617729 9.328044 1.078426e-20 31.89479338  
## conf.high  
## 1 0.1027027  
## 2 198.5873895

Now do a second model with one additional variable (beer).

# As before, we can update the previous model and just write the new formula  
model8 <- update(model7,   
 formula = ill ~ tira +beer)  
  
model8op <- tidy(model8, exponentiate = TRUE, conf.int = TRUE)  
model8op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.06872088 0.4057751 -6.598981 4.139926e-11 0.02820815  
## 2 tira 80.26304344 0.4758675 9.215399 3.101214e-20 33.67736191  
## 3 beer 0.44030520 0.4410228 -1.859965 6.289043e-02 0.18055390  
## conf.high  
## 1 0.1415055  
## 2 221.5079759  
## 3 1.0315458

To compare two models, we will use the **anova** test, which tests for the difference in the residual deviances between the models. This is equivalent to the likelihood ratio test in Stata.

anova(model7, model8, test = "Chisq")

## Analysis of Deviance Table  
##   
## Model 1: ill ~ tira  
## Model 2: ill ~ tira + beer  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)   
## 1 237 154.51   
## 2 236 150.94 1 3.5643 0.05903 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

If the anova test is statistically significant, this suggests that the addition of beer in the model significantly improves the residual deviance of this model.

The results of the anova (p = 0.0590) suggest a borderline significance (at the 0.05 level) for the addition of the variable beer. Remember this might be a confounder, so this may be a sufficient reason for which you may want to keep it in the model regardless of its p-value in the anova test.

Then extend to other variables. Proceed similarly to extend or drop the model according to the anova results.

Keep or drop other variables as needed. Take anova, p-values, magnitude of OR, and the proportion of cases exposed into account in order to decide.

#### Assessing the fit of each model, try to identify the most parsimonious model

Using the AIC function, we obtain the AIC value for each model. You can compare the AIC of multiple models to decide which model is the most parsimonious.

AIC(model7, model8)

## df AIC  
## model7 2 158.5066  
## model8 3 156.9422

You can now add more variables to the model and compare the different AIC; the model with the lowest AIC value will be the most parsimonious.

# Session 2 - logistic regression: including interactions

## Question 9. How would you account for effect modification?

## Help question 9

#### Perform a stratified analysis using logistic regression to check for interactions.

First, let's remember what we saw in the stratified analysis. For this, we first created a 3-way table with exposure, outcome and stratifying variables in that order and then used the epi.2by2 function to obtain the ORs (requires using the case.control method).

a <- table(tira.data.new$beer, tira.data.new$ill, tira.data.new$tira)  
mh <- epi.2by2(a, method = "case.control")  
  
# We can select specific results from mh by subselecting from massoc  
  
# Crude OR  
mh$massoc$OR.crude.wald

## est lower upper  
## 1 0.6071429 0.3492243 1.055546

# Stratum-specific ORs  
mh$massoc$OR.strata.wald

## est lower upper  
## 1 0.9956897 0.2144748 4.6224445  
## 2 0.3205128 0.1162915 0.8833707

# Adjusted OR  
mh$massoc$OR.mh.wald

## est lower upper  
## 1 0.4572593 0.1969943 1.061381

# We can combine all of those elements in to a single table using rbind  
results <- rbind(mh$massoc$OR.crude.wald,   
 mh$massoc$OR.strata.wald,   
 mh$massoc$OR.mh.wald)  
  
  
# We can label the rows of this table as below (this is necessary especially for strata as the strata would otherwise be incorrectly labelled as 1 and 2)  
rownames(results) <- c("Crude", "Strata 0", "Strata 1", "Adjusted")

results

## est lower upper  
## Crude 0.6071429 0.3492243 1.0555465  
## Strata 0 0.9956897 0.2144748 4.6224445  
## Strata 1 0.3205128 0.1162915 0.8833707  
## Adjusted 0.4572593 0.1969943 1.0613814

You can obtain the same ORs using logistic regression:

# No exposure to Tiramisu  
tira0 <- glm(ill ~ beer,   
 data = tira.data.new[tira.data.new$tira == 0,],  
 family = binomial(link = "logit"))  
  
tira0op <- tidy(tira0, exponentiate = TRUE, conf.int = TRUE)  
tira0op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.05194805 0.5127934 -5.767451767 8.047911e-09 0.01584208  
## 2 beer 0.99568966 0.7832563 -0.005515003 9.955997e-01 0.19004231  
## conf.high  
## 1 0.1248666  
## 2 4.6834923

# Exposure to Tiramisu  
tira1 <- glm(ill ~ beer,   
 data = tira.data.new[tira.data.new$tira == 1,],  
 family = binomial(link = "logit"))  
  
tira1op <- tidy(tira1, exponentiate = TRUE, conf.int = TRUE)  
tira1op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 6.5000000 0.3797769 4.928689 8.278316e-07 3.2738445  
## 2 beer 0.3205128 0.5172656 -2.199707 2.782767e-02 0.1122794  
## conf.high  
## 1 14.8056110  
## 2 0.8716553

Add an interaction term to the model. This can be generated directly in the model as below. This variable equals one if tira and beer are present at the same time. Otherwise it is zero.

# Check for interaction between beer and tira  
tirabeer <- glm(ill ~ beer\*tira,   
 data = tira.data.new,  
 family = binomial(link = "logit"))  
  
tirabeerop <- tidy(tirabeer, exponentiate = TRUE, conf.int = TRUE)  
tirabeerop

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.05194805 0.5127934 -5.767451767 8.047911e-09 0.01584208  
## 2 beer 0.99568966 0.7832563 -0.005515003 9.955997e-01 0.19004231  
## 3 tira 125.12499910 0.6381127 7.568119633 3.786652e-14 39.82751065  
## 4 beer:tira 0.32190032 0.9386451 -1.207605918 2.271989e-01 0.05057842  
## conf.high  
## 1 0.1248666  
## 2 4.6834923  
## 3 504.8496932  
## 4 2.1848867

the model is:  
odds = exp(α + β1Χ1 + β2Χ2 + β3Χ3) = cons \* exp(β1tira + β2beer + β3tira\_beer)

The odds of illness among those who consumed tiramisu but did not drink beer was 125.12 times higher compared to those who consumed neither tiramisu nor beer (exposed group: those who consumed tiramisu and did not drink beer, unexposed group=those who were not exposed to tiramisu nor beer).

The odds of illness among those who drank beer but did not consume tiramisu was almost the same compared to those who consumed neither tiramisu nor beer (OR=0.99).

The odds of illness among those who drank beer and consumed tiramisu was 40 times higher compared to those who consumed neither tiramisu nor beer.

In Stata, you would normally use the **lincom** command, but in R, we will use the **glht** function from the **multcomp** package.

#We can use names to extract the coefficient names   
names(coef(tirabeer))

## [1] "(Intercept)" "beer" "tira" "beer:tira"

# linfct specifies the required combination: In this case we want beer and tira and beer:tira=0  
# The odds of illness among those who drank beer and consumed tiramisu compared to those who consumed neither tiramisu nor beer  
  
a <- summary(glht(tirabeer, linfct = c("beer + tira + beer:tira = 0")))  
  
ci <- confint(a)

# Put together (cbind) a table with the exponent of the coefficients and CI, and p-value  
table\_interact <- round(cbind(OR = exp(coef(a)),  
 Interval = exp(ci$confint),  
 Pvalue = a$test$pvalues),  
 digits = 3)  
  
table\_interact

## OR Estimate lwr upr Pvalue  
## beer + tira + beer:tira 40.104 40.104 11.862 135.591 0

#### 

#### Does the interaction term improve the fit of the model?

# Run the model without any interaction between beer and tira  
nointeract <- glm(ill~ tira + beer,   
 data = tira.data.new,  
 family = binomial(link = "logit"))  
  
  
# Run the model with an interaction between beer and tira  
interact <- glm(ill~ tira\*beer,   
 data = tira.data.new,  
 family = binomial(link = "logit"))  
  
  
# Check the fit of the models  
anova(nointeract, interact, test = "Chisq")

## Analysis of Deviance Table  
##   
## Model 1: ill ~ tira + beer  
## Model 2: ill ~ tira \* beer  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1 236 150.94   
## 2 235 149.53 1 1.407 0.2356

#### 

#### Is the model with the interaction a better model?

AIC(nointeract, interact)

## df AIC  
## nointeract 3 156.9422  
## interact 4 157.5352

# Optional Session 3 - Binomial regression: dealing with RRs

## Question 10. If you wanted to use risk ratios, how would you account for the effect of the different exposures?

Start with the simplest model with one exposure variable only

* add one variable at a time and compare models

# Binomial regression with one independent variable  
bin1 <- glm(ill ~ tira,   
 data = tira.data.new,  
 family = binomial(link = "log"))  
  
bin1op <- tidy(bin1, exponentiate = TRUE, conf.int = TRUE)  
bin1op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.04929577 0.3685129 -8.167738 3.142253e-16 0.02148392  
## 2 tira 16.10309272 0.3721283 7.467884 8.149447e-14 8.44137990  
## conf.high  
## 1 0.09314264  
## 2 37.14251394

# With two independent variables  
bin2 <- update(bin1,  
 formula = ill ~ tira + beer)  
  
bin2op <- tidy(bin2, exponentiate = TRUE, conf.int = TRUE)  
bin2op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.05428233 0.3712576 -7.847802 4.233904e-15 0.0235677  
## 2 tira 15.94860691 0.3722222 7.440103 1.006068e-13 8.3628640  
## 3 beer 0.78467280 0.1222724 -1.983183 4.734704e-02 0.5970408  
## conf.high  
## 1 0.1032211  
## 2 36.7828761  
## 3 0.9746920

# Then test for the difference in the 2 models using the anova function  
anova(bin1, bin2, test = "Chisq")

## Analysis of Deviance Table  
##   
## Model 1: ill ~ tira  
## Model 2: ill ~ tira + beer  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)   
## 1 237 154.51   
## 2 236 149.64 1 4.8677 0.02736 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

You can now add three variables to your model.

# The standard approach would be to add the 3rd variable as normal  
bin3 <- glm(ill ~ tira + beer + mousse,  
 data = tira.data.new,  
 family = binomial(link = "log"))

However, this model does not converge and requires specification of starting values for each coefficient in the model (i.e. for the intercept, tira, beer and mousse). The coefficients of the intercept, tira and beer from the bin2 model can be used as the starting points for those coefficients and 0 can be used for mousse (see below).

# Here we save the values of the coefficients of the bin2 model  
coefini = coef(glm(ill ~ tira + beer,  
 data = tira.data.new,  
 family = binomial(link = "log")))  
  
# In bin3, you use the coefficents from bin2 as starting points to facilitate converging of the model  
bin3 <- glm(ill ~ tira + beer + mousse,  
 data = tira.data.new,  
 family = binomial(link = "log"),  
 start = c(coefini,0))  
  
bin3op <- tidy(bin3, exponentiate = TRUE, conf.int = TRUE)  
bin3op

## term estimate std.error statistic p.value conf.low  
## 1 (Intercept) 0.05275242 0.3716952 -7.915479 2.463024e-15 0.0228249  
## 2 tira 14.69261748 0.3827724 7.020739 2.206974e-12 7.2351382  
## 3 beer 0.81761852 0.1221491 -1.648472 9.925590e-02 0.6111628  
## 4 mousse 1.12659426 0.1359679 0.876671 3.806653e-01 0.8704384  
## conf.high  
## 1 0.1008569  
## 2 34.7951145  
## 3 1.0267517  
## 4 1.6898429

# Assessing the fit of each model  
AIC(bin1,bin2,bin3)

## df AIC  
## bin1 2 158.5066  
## bin2 3 155.6389  
## bin3 4 157.0584

You can now check to see if an interaction between tira and beer improves the model. The model that includes an interaction will also not converge without specifying starting points. The starting values added below were obtained from the simpler model without the interaction term. As before, a value of 0 was used as the starting point for the new component (interaction term) of the model.

# check if the interaction improves the model   
nointeract\_binom <- glm(ill~ tira + beer,   
 data = tira.data.new,  
 family = binomial(link = "log"))  
  
  
interact\_binom <- glm(ill~ tira\*beer,   
 data = tira.data.new,  
 family = binomial(link = "log"),  
 start = c(-2.9136, 2.7694, -0.2425, 0))  
  
  
anova(nointeract\_binom, interact\_binom, test = "Chisq")

## Analysis of Deviance Table  
##   
## Model 1: ill ~ tira + beer  
## Model 2: ill ~ tira \* beer  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1 236 149.64   
## 2 235 149.53 1 0.10362 0.7475

You can identify the most parsimonious model using the AIC.

# Assessing the fit of each model  
AIC(nointeract\_binom, interact\_binom)

## df AIC  
## nointeract\_binom 3 155.6389  
## interact\_binom 4 157.5352