Logistic regression

WiSe23/24

Daniela Palleschi

2023-10-11

Table of contents

Learning Objectives

Today we will learn…

- how to model binomial data with logistic regression
- how to interpret log-odds and odds ratio
- how to report a logistic regression model

Resources

- this lecture covers chapter 12 'Genearlised Lienar Models 1: Logistic Regression' (Winter, 2019)
  - we're skipping a few chapters, which I encourage you to go through on your own
  - they cover topics that you presumably have covered in previous courses (namely significance testing, t-values and p-values).

Set-up environment

```
# suppress scientific notation
options(scipen=999)
options(pillar.sigfig = 5)
```

```
# load libraries
pacman::p_load(
               tidyverse,
               here,
```

```
                 broom,
                 janitor,
                 languageR)
```

```
# set preferred ggplot2 theme
theme_set(theme_bw() + theme(plot.title = element_text(size = 10)))
```

Generalised linear models

- linear regression assumes a normal distribution
  - Equation 1, where μ and σ correspond to the mean and standard deviation
- logistic regression assumes a binomial distribution (a.k.a., Bernoulli distribution)
  - Equation 2, where N and p correspond to the number of trials and the probability of y being 1 or 0

$$y \sim Normal(\mu, \sigma) \tag{1}$$
$$y \sim binomial(N = 1, p) \tag{2}$$

- logistic regression is a type of genearlised linear model (GLM)
  - used to model binomial response data

Log-odds, odds ratio, and probabilities

- logistic regression describes the *probability* (p) of observing one outcome or another as a function of a predictor variable
  - e.g., the absence or presence of some phenomenon (word order, schwa, etc.) or button responses (yes/no, accept/reject)
- this can be described as the probability, odds, or log-odds of a particular outcome over another

Probability

- probability ranges from 0 (no chance) to 1 (certain)
  - 50% chance = probability of 0.5

Odds (ratio)

- odds range from 0 to infinity
  - *the odds that I'll win are 2:1* ($\frac{2}{1} = 2$ in favour of my winning)
  - *the odds that you'll win are 1:2*($\frac{1}{2} = 0.5$)
  - if the odds are even (1:1), then: $\frac{1}{1} = 1$
- odds of 1 correspond to a probability of 0.5

Log-odds

- log-odds are just the logarithmically-transformed odds
  - $log(2) =$ 0.6931472
  - $log(0.5) =$ -0.6931472
  - $log(1) = 0$
- so the log-odds of 0 correspond to a probability of 0.5 (and odds of 1)

Calculating odds/log-odds/probability

- Equations 3-5 demonstrate the relationship between the three

$$p = \frac{odds}{1 + odds} \tag{3}$$

$$odds = \frac{p}{1 - p} \tag{4}$$

$$log\ odds = exp(odds) \tag{5}$$

- TASK: Using R and Equations 3-5, compute:
  - the probability and log odds for odds of 0.082
  - the log odds and odds for a probability of 0.924
  - the probability and odds for a log odds of -2.5

Comparing odds/log-odds/probability

- Table 1 gives an example of how the three relate to each other
  - did you get the correct values?
- try the task again, this time using `plogis()`, which produces a probility from a log odds

Table 1: Comparison of different values of probabilities/odds/log-odds

prob	0.007	0.023	0.076	0.223	0.5	0.777	0.924	0.977	0.993
odds	0.007	0.024	0.082	0.287	1.0	3.490	12.182	42.521	148.413
log_odds	-5.000	-3.750	-2.500	-1.250	0.0	1.250	2.500	3.750	5.000

Plotting odds/log-odds/probability

- this relationship is demonstrated in Figure 1
- take your time to really understand these plots

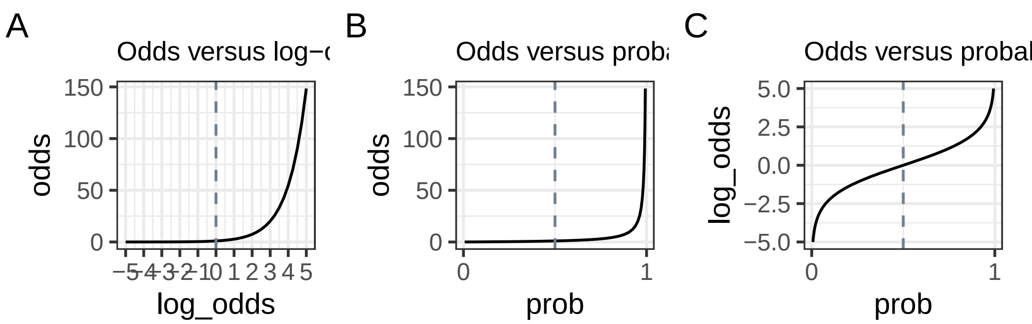


Figure 1: Relationship between probability, odds, and log-odds

Logistic regression

- let's run our first logistic regression to understand this relationship better
- Most relevant to the output of a logistic regression model is Figure 1 C
    - the model will output log-odds, and we most likely want to interpret them in terms of probabilities

Load data

- load in the Biondo et al. (2022) dataset again
    - let's look at the binomial measure *regression in* at the *verb* region

```
df_tense <-
  read_csv(here("data", "Biondo.Soilemezidi.Mancini_dataset_ET.csv"),
           locale = locale(encoding = "Latin1") # for special characters in Spanish
           ) |>
```

```
  clean_names() |>
  mutate(gramm = ifelse(gramm == "0", "ungramm", "gramm"))  |>
  filter(roi == 4,
         adv_type == "Deic")
```

EDA

- conduct a quick EDA: print head of data

```
head(df_tense)
```

```
# A tibble: 6 x 13
  sj     item adv_type adv_t  verb_t gramm     roi label    fp    gp    tt    ri
  <chr> <dbl> <chr>    <chr>  <chr>  <chr>   <dbl> <chr> <dbl> <dbl> <dbl> <dbl>
1 1        54 Deic     Past   Past   gramm       4 enca~  1027  1027  1027     0
2 1         4 Deic     Future Future gramm       4 cole~   562   562  1337     1
3 1        62 Deic     Past   Past   gramm       4 esqu~   293  1664  1141     0
4 1        96 Deic     Future Past   ungramm     4 cons~   713  1963  1868     0
5 1        52 Deic     Past   Past   gramm       4 desa~   890   890  1707     1
6 1        90 Deic     Future Past   ungramm     4 dece~   962   962   962     0
# i 1 more variable: ro <dbl>
```

- and summary

```
df_tense |>
  select(roi, ri, ro) |>
  summary()
```

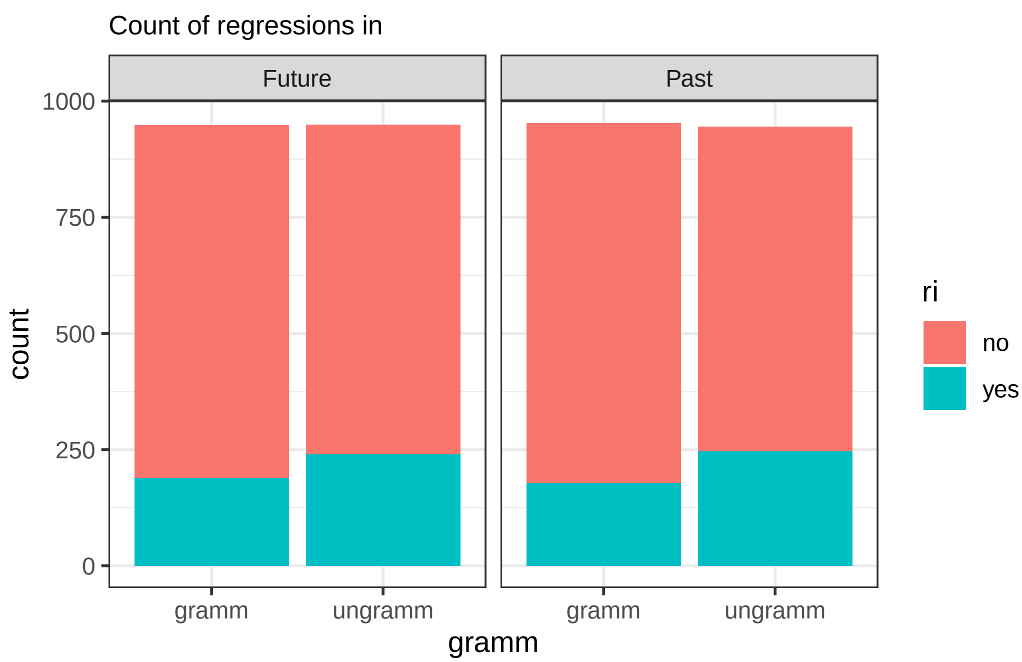
```
      roi          ri               ro         
 Min.   :4   Min.   :0.0000   Min.   :0.00000  
 1st Qu.:4   1st Qu.:0.0000   1st Qu.:0.00000  
 Median :4   Median :0.0000   Median :0.00000  
 Mean   :4   Mean   :0.2248   Mean   :0.08169  
 3rd Qu.:4   3rd Qu.:0.0000   3rd Qu.:0.00000  
 Max.   :4   Max.   :1.0000   Max.   :1.00000  
             NA's   :45       NA's   :45       
```

Plot

- let's plot the count of yes/no for regressions in

```
facet_labels <-
  c(
    "ri" = "Reg. In",
    "ro" = "Reg. Out",
    "Future" = "Future",
    "Past" = "Past"
  )

# fig_reg <-
  df_tense |>
  filter(roi == "4") |>
  mutate(gramm = as_factor(gramm),
         ri = ifelse(ri == 1, "yes", "no")) |>
  drop_na(ri) |>
  ggplot() +
  labs(title = "Count of regressions in") +
  aes(x = gramm, fill = ri) +
  geom_bar() +
  facet_grid(.~verb_t, labeller = as_labeller(facet_labels))
```



Model

- run our model
  - `verb_t` and `gramm` are each two-level factors: set sum coding
  - `past` and `grammatical` = −0.5, and `future` and `ungrammatical` = `+0.5`

Contrast coding

- for `verb_t`

```
# verb_t as factor
df_tense$verb_t <- as.factor(df_tense$verb_t)
# check levels/order
levels(df_tense$verb_t)
```

```
[1] "Future" "Past"
```

```
# set contrasts accordingly
contrasts(df_tense$verb_t) <- c(+0.5, -0.5)
# check contrasts
contrasts(df_tense$verb_t)
```

```
       [,1]
Future  0.5
Past   -0.5
```

- for `gramm`

```
# as factor
df_tense$gramm <- as.factor(df_tense$gramm)
# check
levels(df_tense$gramm)
```

```
[1] "gramm"   "ungramm"
```

```
# set contrasts
contrasts(df_tense$gramm) <- c(-0.5, +0.5)
# check contrasts
```

term	estimate	std.error	statistic	p.value
(Intercept)	-1.25	0.04	-31.81	0.00
verb_t1	0.02	0.08	0.27	0.79
gramm1	0.37	0.08	4.68	0.00
verb_t1:gramm1	-0.12	0.16	-0.76	0.45

```
contrasts(df_tense$gramm)
```

```
        [,1]
gramm   -0.5
ungramm  0.5
```

Fit model

- we use the `glm()` function to fit a *genearlised* linear model
  - use the argument `family = "binomial"` to indicate our data are binomial

```
fit_tense_ri <-
  glm(ri ~ verb_t*gramm,
    data = df_tense,
    family = "binomial")
```

Coefficients

```
tidy(fit_tense_ri) %>%
  mutate(p.value = as.numeric(p.value)) |>
  mutate(p.value = round(p.value,10)
         ) |>
  knitr::kable(digits = 2) |>
  kableExtra::kable_styling()
```

- the intercept is negative: below 0
  - verb tense is positive: more regressions in for the `future` compared to the `past`, holding grammaticality constant
  - grammaticality is positive: more regressions in for the `ungrammatical` than `grammatical` conditions

Interpreting 0

- what does zero mean here?
  - logistic regression gives the estimates in log-odds
  - in log-odds, a value of 0 means there is an equal probability of a regression in or out for both conditions (as in Table 1)
  - i.e., the slope is flat (or not significantly different from 0)
- How can we convert our log-odds estimates to something more interpretable, like probabilities?

Log-odds to probabilities

- we can just use the `plogis()` function
- we can also just use the `exp()` function to get the odds ratio from the log-odds
- let's look at our coefficients in probabilities:

```
plogis(-1.23) # intercept in prob
```

```
[1] 0.2261814
```

```
plogis(0.0277) # tense in prob
```

```
[1] 0.5069246
```

- and in odds

```
exp(-1.23) # intercept in odds
```

```
[1] 0.2922926
```

```
exp(0.0277) # tense in odds
```

```
[1] 1.028087
```

term	estimate	std.error	statistic	p.value	prob	odds
(Intercept)	-1.25	0.04	-31.81	0.00	0.22	0.29
verb_t1	0.02	0.08	0.27	3.16	0.51	1.02
gramm1	0.37	0.08	4.68	0.00	0.59	1.44
verb_t1:gramm1	-0.12	0.16	-0.76	1.78	0.47	0.89

Streamlining

- this is a bit tedious
  - we can also just feed a tibble column through the `plogis()` and `exp()` functions to print a table with the relevant probabilities and odds

```
tidy(fit_tense_ri) %>%
  mutate(p.value = round(p.value*4,10),
         prob = plogis(estimate),
         odds = exp(estimate)
         ) |>
  mutate_if(is.numeric, round, 4) |>
  knitr::kable(digits = 2) |>
  kableExtra::kable_styling()
```

Interpreting our slopes

- the odds of a regression in for the future tense versus the past tense is ~1, with the corresponding probability of 0.51 _ unsuprisingly, we see this p-value indicates this effect was not significant ($p > .05$), and the z-value (note: not t-value!) is also low
  - z-values correspond to the estimate divided by the standard error; it's interpretation is similar to that of the t-value: a z-value of ~2 or higher will likely have a p-value below 0.05.

Interpreting interaction

- interaction term is negative, what does this mean?
  - the effect of congruence is different in either level of tense
  - these effects are often more easily interpreted with a visualisation, e.g., using the `plot_model()` function from the `sjPlot` package (this effect is not significant, however)

term	estimate	std.error	statistic	p.va
(Intercept)	-1.25	0.04	-31.81	0
verb_t1	0.02	0.08	0.27	3
gramm1	0.37	0.08	4.68	0
verb_t1:gramm1	-0.12	0.16	-0.76	1

```r
sjPlot::plot_model(fit_tense_ri,
                   type = "eff",
                   terms = c("gramm", "verb_t")) +
  geom_line() +
  theme(text = element_text(size = 26))
```

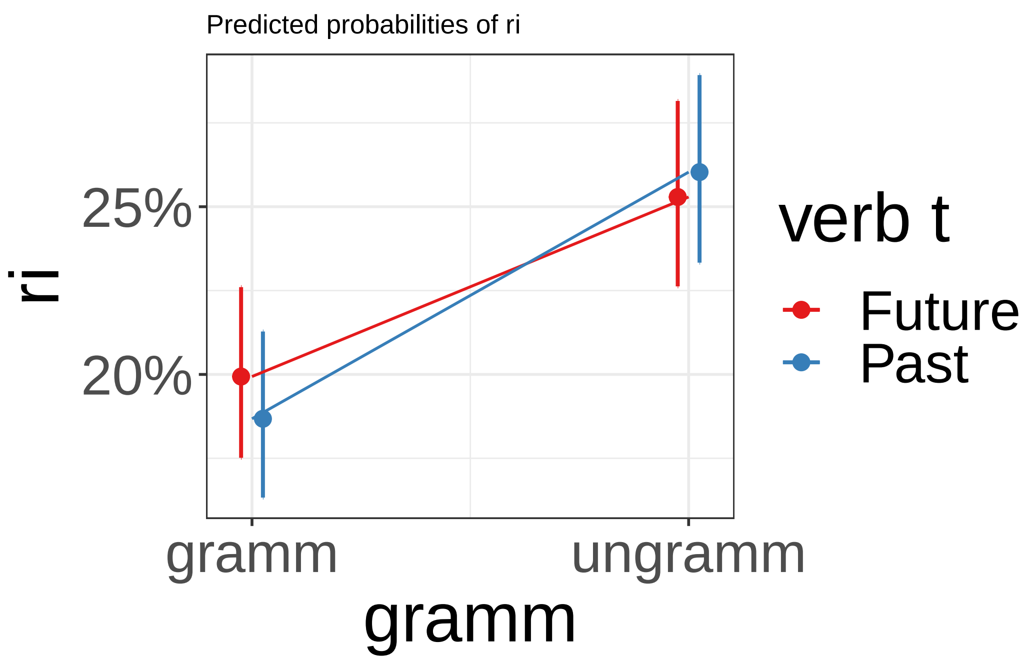


Figure 2: Interaction plot of grammaticality and tense

Extracting predicted values

- we can use the `predict()` function to extract the predicted values for each condition
- We could just simply print the predicted values (`predict(fit_tense_ri)`), append the predicted values to the data frame

```
# make sure dataset is the same length as the model data
df_tense_v <-
  df_tense |>
  drop_na(ri)

# append model estimates
df_tense_v <-
  augment(fit_tense_ri, data = df_tense_v) |>
  distinct(verb_t, gramm, .keep_all = T) |>
  arrange(verb_t) |>
  select(verb_t, gramm, .fitted)
```

Predicted values and slopes

- now if we look at the predicted log-odds values for the future and past tenses:

```
df_tense_v |>
  summarise(
    mean_tense = mean(.fitted),
    .by = verb_t)
```

```
# A tibble: 2 x 2
  verb_t mean_tense
  <fct>       <dbl>
1 Future    -1.2367
2 Past      -1.2577
```

- What is the difference between these two numbers (in our model summary)?
  - it's 0.03: our slope for `verb_t`

```
df_tense_v |>
  summarise(
    mean_gramm = mean(.fitted),
    .by = gramm)
```

```
# A tibble: 2 x 2
  gramm   mean_gramm
  <fct>        <dbl>
1 gramm      -1.4307
2 ungramm    -1.0638
```

- What is the difference between these two numbers (in our model summary)?
  - it's 0.32: our slope for `verb_t`
- slopes for `verb_t` and `gramm` correspond to the predicted difference between their levels

Interpreting our coefficients

- what do our estimates reflect, though?
  - let's remind ourselves of the rate of regressions in at the verb region:

```
intercept <- tidy(fit_tense_ri)$estimate[1]
tense <- tidy(fit_tense_ri)$estimate[2]
gramm <- tidy(fit_tense_ri)$estimate[3]
interact <- tidy(fit_tense_ri)$estimate[4]
```

- let's remind ourselves of our contrast coding, so we can plug these into our equation of a line

```
contrasts(df_tense_v$verb_t)
```

```
       [,1]
Future  0.5
Past   -0.5
```

```
contrasts(df_tense_v$gramm)
```

```
        [,1]
gramm   -0.5
ungramm  0.5
```

Calculating our predictions

- what's the probability of a regression in for the past (`tense` = -0.5) grammatical (`gramm` = -0.5) condition?

```
plogis(intercept + tense*-.5 + gramm*-.5)
```

```
[1] 0.1913675
```

- and past ungrammatical (change `gramm` to +0.5)?

```
plogis(intercept + tense*-.5 + gramm*.5)
```

```
[1] 0.2545957
```

- And for the future condition (`verb_t` = 0.5) grammatical (`gramm` = -0.5)?

```
plogis(intercept + tense*.5 + gramm*-.5)
```

```
[1] 0.1946325
```

- and future ungrammatical (gramm = +0.5)?

```
plogis(intercept + tense*.5 + gramm*.5)
```

```
[1] 0.2585946
```

$$y_i = b_0 + b_1 x_i + b_2 x_1 + ... + e \tag{6}$$

Math with factors

- so, even when our dependent *and* independent variables are categorical, we can include them in our equation of a line (equation 6)
- we do this by assigning them numerical values
  - a probability/log odd/odds for a binomial dependent variable
  - and contrast coding for categorical predictors

Table 2: Model summary for regressions in at the verb region. Estimates are given in log odds.

Predictor	b	95% CI	z	p
Intercept	-1.25	[-1.32, -1.17]	-31.81	$<$ .001
Verb t1	0.02	[-0.13, 0.17]	0.27	.789
Gramm1	0.37	[0.21, 0.52]	4.68	$<$ .001
Verb t1 $\times$ Gramm1	-0.12	[-0.43, 0.19]	-0.76	.445

Reporting

Sonderegger (2023) (Section 6.9) makes a few important points regarding coefficients:

> Reporting a logistic regression model in a write-up is generally similar to reporting a linear regression model: the guidelines and rationale in section 4.6 for reporting individual coefficients and the whole model hold, with some adjustments.
>
> For each regression coefficient you report at a minimum the coefficient estimate, its SE, the test statistic value…and corresponding p-value.
>
> As for linear regression, it is useful to also give visualizations, CIs, and basic descriptive statistics, but what is appropriate will depend on context and space.
>
> Model prediction plots are especially important for interpreting logistic regressions, as discussed in section 6.7.3.

Producing table summaries with `papaja`

- we can produce such a table using e.g., `papaja` package (true for any type of model; **?@tbl-glm-summary**)

```
library(papaja)

fit_tense_ri |>
  apa_print() |>
  apa_table(label = "tbl-glm-summary",
            caption = "Model summary for regressions in at the verb region. Estimates are
```

Table 3: Same table with ‘tidy()‘

term	estimate	std.error	prob	statistic	p.value	conf.low	conf.high
(Intercept)	-1.25	0.04	0.22	-31.81	0.00	-1.32	-1.17
verb_t1	0.02	0.08	0.51	0.27	0.79	-0.13	0.17
gramm1	0.37	0.08	0.59	4.68	0.00	0.21	0.52
verb_t1:gramm1	-0.12	0.16	0.47	-0.76	0.45	-0.43	0.19

term	description/other terms

Producing table summaries with `broom::tidy()`

- or by extracting the model summary with `tidy()`, and even adding our probabilities (**?@tbl-glm-summary-tidy**)

```
tidy(fit_tense_ri, conf.int = TRUE) |>
  mutate(prob = plogis(estimate)) |>
  relocate(prob, .after = std.error) |>
  apa_table(label = "tbl-glm-summary-tidy",
            caption = "Same table with `tidy()`")
```

Learning Objectives

Today we learned…

- how to model binomial data with logistic regression
- how to interpret log-odds and odds ratio
- how to report a logistic regression model

Important terms

Task

Regressions out

Using the same dataset, run a logistic model exploring regressions in (`ri`) at the adverb region (`roi = "2"`). Before you run the model, do you have any predictions? Try plotting the regressions in for this region first, and generate some summary tables to get an idea of the distributions of regressions in across conditions.

Dutch verb regularity

Load in the `regularity` data from the `languageR` package.

```
df_reg <-
  regularity |>
  clean_names()
```

> Regular and irregular Dutch verbs and selected lexical and distributional properties.

Our relevant variables will be:

- `written_frequency`: a numeric vector of logarithmically transformed frequencies in written Dutch (as available in the CELEX lexical database).
- `regularity`: a factor with levels irregular (1) and regular (0).
- `verb`: a factor with the verbs as levels.

1. Fit a logistic regression model to the data which predicts verb regularity by written frequency. Consider: What type of predictor variable do you have, and what steps should you take before fitting your model?

2. Print the model coefficients, e.g., using `tidy()`.

3. Interpret the coefficients, either in log-odds or probabilities. Report your findings.

Literaturverzeichnis

Biondo, N., Soilemezidi, M., & Mancini, S. (2022). Yesterday is history, tomorrow is a mystery: An eye-tracking investigation of the processing of past and future time reference during sentence reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *48*(7), 1001–1018. https://doi.org/10.1037/xlm0001053

Sonderegger, M. (2023). *Regression Modeling for Linguistic Data*.

Winter, B. (2019). Statistics for Linguists: An Introduction Using R. In *Statistics for Linguists: An Introduction Using R*. Routledge. https://doi.org/10.4324/9781315165547