Week 07: GLMs and logistic regression

Binary outcomes, links, odds ratios, predicted probabilities

Bartosz Maćkiewicz

Generalized linear models

Generalized linear models extend ordinary least squares regression to outcomes that are not well described by a normal distribution with constant variance.

The basic idea is familiar: we still describe an outcome as a function of predictors. What changes is the assumed distribution of the outcome and the scale on which the predictors act. Instead of forcing everything into `lm()`, we choose a model whose assumptions match the response variable.

What changes in a GLM?

A generalized linear model has three pieces:

1. a **random component**, which describes the distribution of the outcome,
2. a **linear predictor**, which combines predictors as a weighted sum,
3. a **link function**, which connects the linear predictor to the expected outcome.

For logistic regression, the outcome is binary, the random component is binomial, and the link function is the **logit**.

Logistic regression

So far, many of our examples involved measured outcomes: reaction times, scores, prestige ratings, completion times. In many cognitive and behavioral studies, however, the outcome has only two possible values:

- correct versus incorrect,
- success versus failure,
- regular versus irregular,
- selected versus not selected.

For such data, we want to model the **probability** of one outcome as a function of the predictors.

Why ordinary regression is a poor fit

The first problem is that probabilities are bounded. A probability below `0` or above `1` is not a bad prediction in the usual sense; it is not a probability at all.

An ordinary linear model does not know about this boundary. If we code the outcome as `0` and `1`, `lm()` can still draw a straight line, but the fitted line is not constrained to stay inside the probability scale.

Another problem: variance changes

Binary data also have a mean-variance relationship. When the probability is near `0` or near `1`, there is little room for variation. When the probability is near `0.5`, the outcome varies the most.

That is different from the constant-variance assumption behind ordinary least squares. It is one reason why a model built for continuous outcomes is not a natural model for binary responses.

Why proportions are not enough

It is tempting to collapse binary responses into proportions and analyze those proportions directly. But proportions do not carry all the information unless we also know how many observations produced them.

A proportion of `0.75` based on `4` observations and a proportion of `0.75` based on `400` observations should not be treated as equally precise. Logistic regression keeps the binary observations and uses the appropriate binomial likelihood.

The logistic solution

Logistic regression models probabilities indirectly. It does not make the predictors act directly on p. Instead, it models the **logit** of the probability:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$$

The quantity $p/(1-p)$ is the **odds**. The logit is the log of those odds.

The zero point of the logit scale

A logit of `0` is the neutral point of the scale.

$$\text{logit}(p) = 0 \quad \iff \quad \frac{p}{1-p} = 1 \quad \iff \quad p = .50$$

So a positive logit means that the outcome is more likely than not. A negative logit means that the outcome is less likely than not.

From logits back to probabilities

The logit scale is useful for modeling because it is unbounded: logits can range from $-\infty$ to $+\infty$. That gives the linear predictor room to behave like an ordinary regression equation.

After fitting the model, we can return to the probability scale with the inverse-logit:

$$p = \frac{1}{1 + e^{-\eta}}$$

Here, η is the **linear predictor**: the model's predicted log-odds, such as $\beta_0 + \beta_1 x$.

In R, that inverse-logit function is `plogis()`.

Course example

We will use data from Tobia, Newman, and Knobe's study on natural kind judgments. Participants read short vignettes about gold, tigers, or water.

The experiment crossed two factors:

- `Kind`: gold, tigers, or water,
- `Structure`: same appearance or different appearance.

The outcome we will model is whether the participant chose the strict exclusion response.

A vignette example

Water + same appearance. Suppose that humans travel to another galaxy and land on a planet that looks exactly like Earth. They call it **Twin Earth**. Its lakes and rivers contain a liquid that looks and tastes just like water. The astronauts drink it and use it just as they would use water on Earth.

When they analyze the liquid, however, they discover that it is not composed of H2O. It is made of elemental atoms rather than compound molecules. Scientists conclude that the liquid behaves just like water outside the lab, but does not belong to the same scientific category as the liquid in Earth's lakes and rivers.

The vignette manipulation

In the **same appearance** condition, the target object looked and behaved like the familiar Earth object, but had a different underlying causal property.

In the **different appearance** condition, the target object lacked the familiar superficial properties and also had a different underlying causal property.

The theoretical question is whether people rely only on deeper causal structure, or whether superficial appearance also shapes category judgments.

Response options

Participants then received three statements and were asked to indicate which one they agreed with most. For the water vignette, the question was:

With which of the following do you most agree?

1. The liquid from Twin Earth is water. (*Is a member*)
2. The liquid from Twin Earth is not water. (*Is not a member*)
3. There is a sense in which the liquid from Twin Earth is water, but ultimately it is not truly water. (*Two senses*)

The cleaned dataset has a binary variable, `RatingDich`. Here, `RatingDich == 1` means the participant chose **"Is Not a Member."**

Code: observed proportions

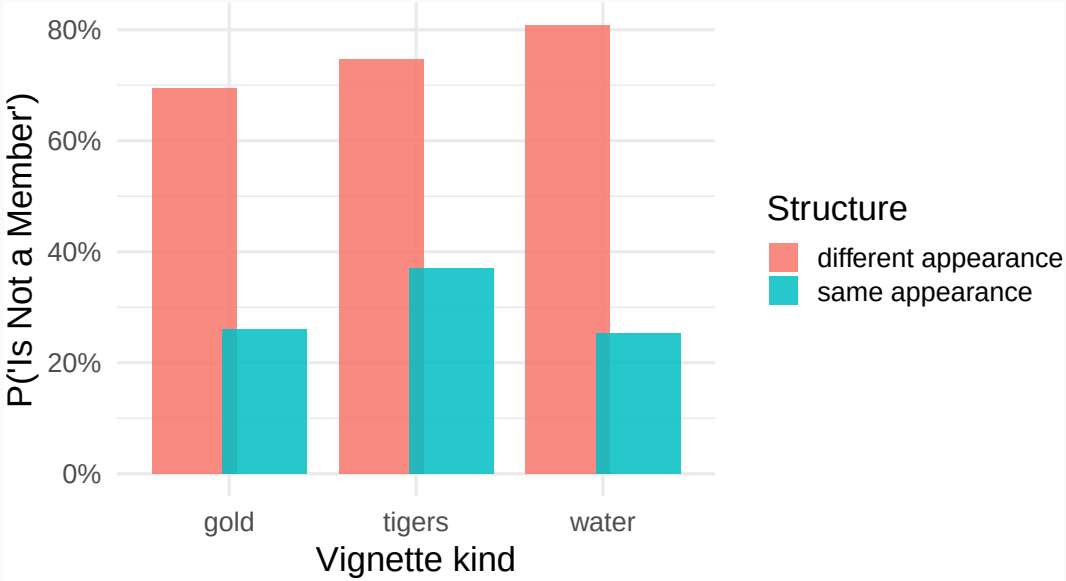
```r
tobia <- read.csv("tobia2019_cleaned.csv")

prop.table(
  table(tobia$Structure, tobia$RatingDich),
  margin = 1
)
```

Output: observed proportions

```
                          0     1
different appearance 0.247 0.753
same appearance      0.706 0.294
```

The probability of the strict exclusion response is much higher in the `different appearance` condition. Before fitting a model, the raw proportions already show that vignette structure matters.

Plot: proportions by vignette



Code: odds and logits

```r
tab <- table(tobia$Structure, tobia$RatingDich)

odds <- tab[, "1"] / tab[, "0"]
logit <- log(odds)

round(cbind(odds, logit), 3)
```

Output: odds and logits

```
                      odds  logit
different appearance 3.056  1.117
same appearance      0.416 -0.877
```

The `different appearance` condition has odds greater than `1`, because the strict exclusion response is more common than the other responses combined. The `same appearance` condition has odds below `1`.

Fitting the first model

```
fit_structure <- glm(
  RatingDich ~ Structure,
  data = tobia,
  family = binomial
)

summary(fit_structure)
```

The argument `family = binomial` is the crucial part. It tells `glm()` that the outcome is binary and that the model should use the logit link.

Output: model summary

```
Call:
glm(formula = RatingDich ~ Structure, family = binomial, data = tobia)

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)                 1.1170     0.1920   5.817 5.99e-09 ***
Structuresame appearance   -1.9936     0.2378  -8.384  < 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: 540.18  on 390  degrees of freedom
Residual deviance: 459.83  on 389  degrees of freedom
AIC: 463.83

Number of Fisher Scoring iterations: 4
```

Reading the summary

The intercept is the log-odds of `RatingDich == 1` in the reference condition. With the default factor ordering, the reference condition is `different appearance`.

The `Structuresame appearance` coefficient is the difference in log-odds between `same appearance` and `different appearance`. It is negative, which means the strict exclusion response becomes less likely in the same-appearance condition.

The printed `z` test asks whether the coefficient differs from zero on the logit scale.

Code: odds ratios

```
broom::tidy(
  fit_structure,
  conf.int = TRUE,
  exponentiate = TRUE
)
```

Exponentiating a logistic-regression coefficient converts it from a log-odds difference into an **odds ratio**. Odds ratios are multiplicative effects on odds, not additive effects on probabilities.

```
# A tibble: 2 x 4
  term              odds_ratio   low  high
  <chr>                  <dbl> <dbl> <dbl>
1 reference odds          3.06  2.12  4.51
2 same vs different       0.136 0.085 0.215
```

The odds ratio for `Structuresame appearance` is far below `1`. This means that the odds of a strict exclusion response are much lower in the `same appearance` condition than in the `different appearance` condition.

Why odds ratios are not enough

Odds ratios are useful and widely reported, but they are often hard to read substantively. An odds ratio tells us how the **odds** change, not how the **probability** changes.

For communication, predicted probabilities are usually clearer. They answer the question readers often have first: how likely is the outcome in each condition?

Code: predicted probabilities

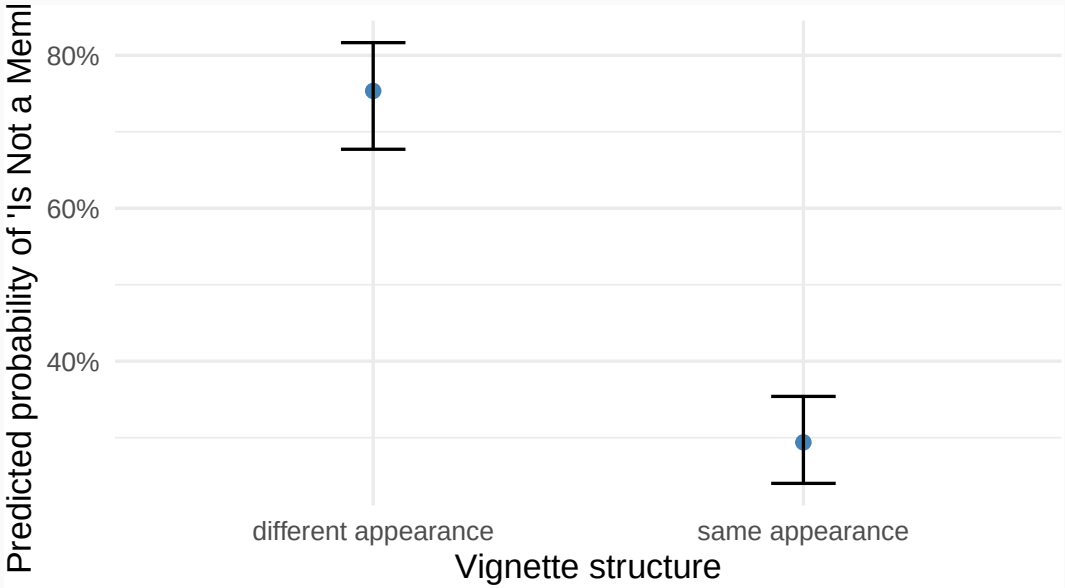
```
newdat <- tibble(Structure = levels(tobia$Structure))

pred <- predict(
  fit_structure, newdata = newdat,
  type = "link", se.fit = TRUE
)

newdat |>
  mutate(
    p = plogis(pred$fit),
    p_low = plogis(pred$fit - 1.96 * pred$se.fit),
    p_high = plogis(pred$fit + 1.96 * pred$se.fit)
  )
```

Output: predicted probabilities

```
# A tibble: 2 x 4
  Structure                p p_low p_high
  <chr>                <dbl> <dbl>  <dbl>
1 different appearance 0.753 0.677  0.817
2 same appearance      0.294 0.24   0.354
```

The confidence interval is computed on the logit scale first, then transformed back to the probability scale. This keeps the interval inside the `0` to `1` range.

Plot: predicted probabilities



Adding `Kind`

The first model asks whether `Structure` matters. The next model asks whether the pattern is the same for gold, tigers, and water.

There are two natural models:

- an **additive** model: `Structure + Kind`,
- an **interaction** model: `Structure * Kind`.

The interaction model allows the effect of `Structure` to differ across the three kinds.

Code: comparing nested models

```
fit_add <- glm(
  RatingDich ~ Structure + Kind,
  data = tobia,
  family = binomial
)

fit_int <- glm(
  RatingDich ~ Structure * Kind,
  data = tobia,
  family = binomial
)

anova(fit_add, fit_int, test = "Chisq")
```

Output: model comparison

```
Analysis of Deviance Table

Model 1: RatingDich ~ Structure + Kind
Model 2: RatingDich ~ Structure * Kind
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       387     457.63
2       385     455.00  2   2.6253   0.2691
```

For GLMs, this is a likelihood-ratio comparison of two nested models. Here it asks whether allowing the `Structure` effect to vary by `Kind` improves the model enough to justify the additional parameters.

Sequential tests with `anova()`

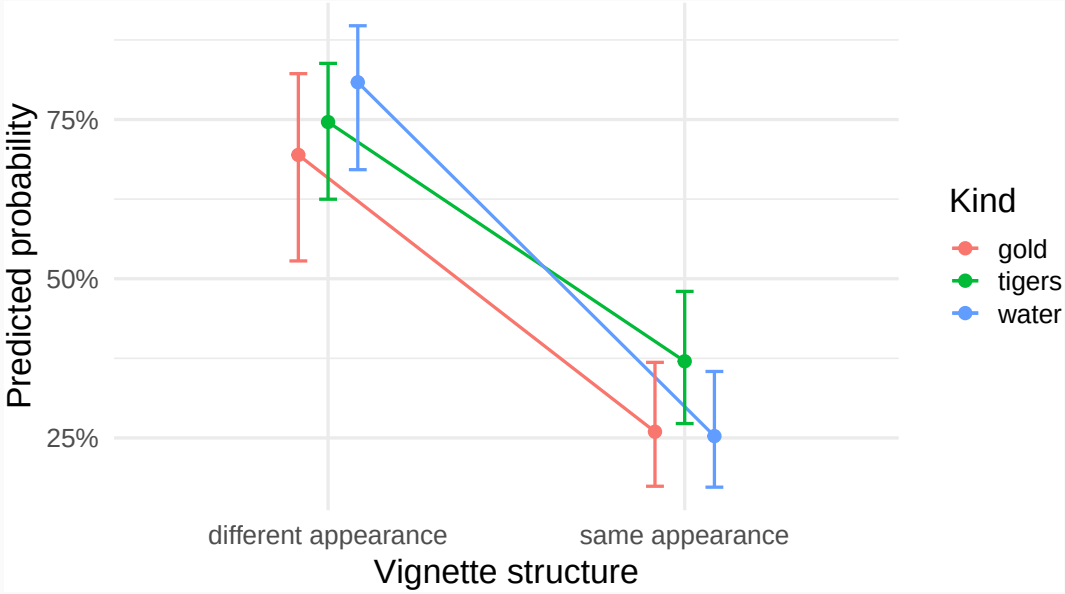
For `glm()` objects, `anova(model, test = "Chisq")` can also be used as a sequential analysis of deviance table. Terms are added in order, and each row asks whether the newly added term explains additional deviance.

This is useful for multi-level factors. A factor like `Kind` has more than two levels, so there is not just one coefficient that represents the whole factor.

Code: predicted probabilities by `Kind`

```
grid <- tidyr::expand_grid(
  Structure = levels(tobia$Structure),
  Kind = levels(tobia$Kind)
)

pred <- predict(
  fit_int,
  newdata = grid,
  type = "link",
  se.fit = TRUE
)

grid <- grid |>
  mutate(p = plogis(pred$fit))
```

Plot: interaction model predictions



Reading the interaction

The plot is the easiest way to read the interaction model. In each kind, the same-appearance condition produces a lower probability of the strict exclusion response than the different-appearance condition.

The size of that difference is not identical across kinds, but the model comparison tells us whether the additional interaction terms are worth keeping.

Overall fit

For binary outcomes, there is no single R^2 with exactly the same interpretation as in ordinary linear regression. Several pseudo-R^2 measures compare the fitted model with a baseline model, but they should not be read as “percent of variance explained.”

Nagelkerke’s R^2 is scaled to run from `0` to `1`, which makes it look familiar. The interpretation is **not** the same as OLS R^2: use it as a rough summary of improvement over a null model, not as a percentage of variance explained.

Code: fit summaries

```
broom::glance(fit_int) |>
  select(nobs, df.residual, deviance, null.deviance, AIC)

performance::r2_nagelkerke(fit_int)
```

Output: fit summaries

```
# A tibble: 1 x 5
   nobs df.residual deviance null.deviance   AIC
  <int>       <int>    <dbl>         <dbl> <dbl>
1   391         385     455.          540.  467.

Nagelkerke's R2
      0.2613973
```

Checking a logistic model

Model checking for logistic regression is not a search for normally distributed residuals. The model is designed for a binary response, so the residuals will not behave like the residuals from `lm()`.

A more useful question is whether the model is **systematically wrong** in some part of the prediction range. Do observations with low predicted probabilities, medium predicted probabilities, or high predicted probabilities look different from what the model expects?

Why raw residual plots are awkward

With binary outcomes, each observation is either `0` or `1`. Residual plots therefore often show bands or repeated points rather than a smooth cloud.

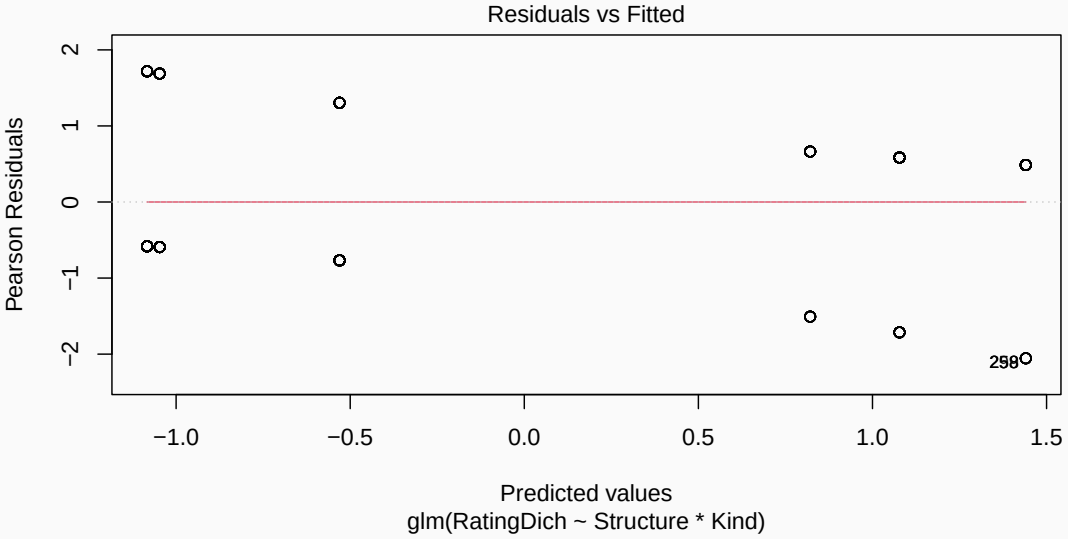
That is especially true here because the predictors are categorical. The interaction model produces only a small number of distinct fitted values: one for each combination of `Kind` and `Structure`.

Code: residuals vs fitted

```r
plot(fit_int, which = 1)
```

`plot()` works because a fitted GLM object also inherits from `lm`. Here, `which = 1` asks only for the residuals-vs-fitted plot.

Plot: residuals vs fitted



Binned residuals

A binned residual plot is often easier to read for logistic regression. Instead of staring at individual `0`/`1` residuals, it groups cases with similar predicted probabilities and plots the **average residual** in each group.

If the model is doing reasonably well, those averages should stay near zero and inside the error bounds. A systematic pattern would suggest that the model is missing something.

Code: binned residuals

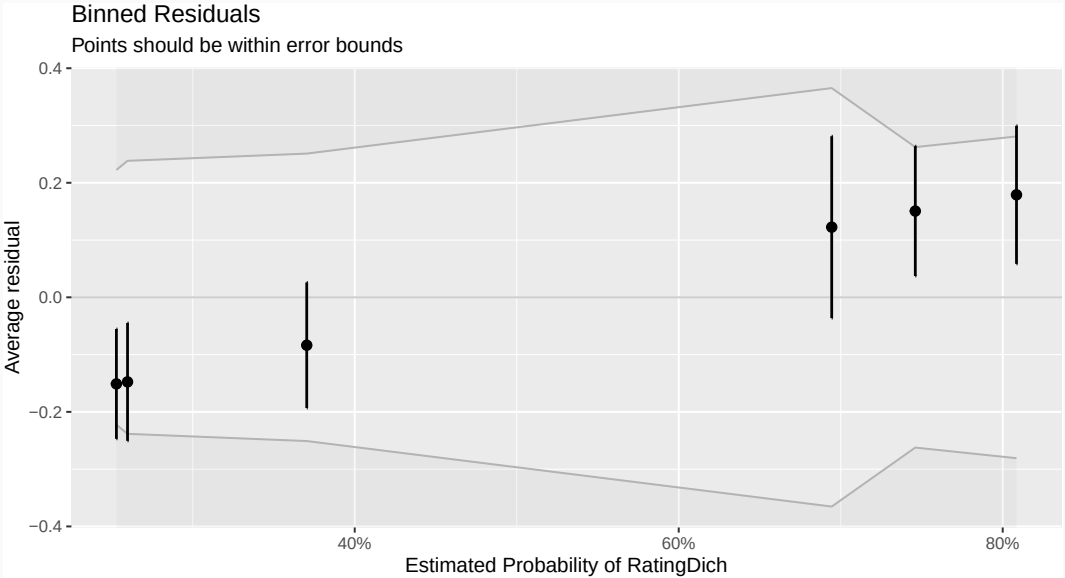
```
binned_res <- performance::binned_residuals(fit_int)

binned_res
plot(binned_res)
```

Output: binned residuals

```
Ok: About 100% of the residuals are inside the error bounds.
```

For this model, the binned residuals do not suggest a large systematic lack of fit.

Plot: binned residuals



Takeaways

- Logistic regression is a GLM for binary outcomes.
- The model is linear on the **logit** scale, not on the probability scale.
- Coefficients can be converted to **odds ratios** with `exp()`.
- Predicted probabilities are usually the clearest way to communicate results.
- Model comparison and diagnostics still matter, but their tools differ from `lm()`.