Tutorial 11: Categorical Data

```
#| edit: false
#| output: false
webr::install("gradethis", quiet = TRUE)
library(gradethis)
options(webr.exercise.checker = function(
  label, user_code, solution_code, check_code, envir_result, evaluate_result,
  envir_prep, last_value, engine, stage, ...
) {
  if (is.null(check_code)) {
    # No grading code, so just skip grading
    invisible(NULL)
  } else if (is.null(label)) {
    list(
      correct = FALSE,
      type = "warning",
      message = "All exercises must have a label."
    )
  } else if (is.null(solution_code)) {
    list(
      correct = FALSE,
      type = "warning",
      message = htmltools::tags$div(
        htmltools::tags$p("A problem occurred grading this exercise."),
        htmltools::tags$p(
          "No solution code was found. Note that grading exercises using the ",
          htmltools::tags$code("gradethis"),
          "package requires a model solution to be included in the document."
        )
      )
    )
  } else {
    gradethis::gradethis_exercise_checker(
```

```
      label = label, solution_code = solution_code, user_code = user_code,
      check_code = check_code, envir_result = envir_result,
      evaluate_result = evaluate_result, envir_prep = envir_prep,
      last_value = last_value, stage = stage, engine = engine)
  }
})
```

Q1 - Proportion of fruity candies

For this tutorial, we will be using a Halloween candy dataset that compares different popular candy brands. Calculate the proportion of fruity candies from the dataset.

Photo by Joanna Kosinska on Unsplash

> **Info**
>
> With categorical datsets like this one, information is usually stored in the form of indicator variables, i.e. value=1 means yes and value=0 means no. For such a 0/1 indicator, the proportion of 1s is mean(indicator == 1).

> **Preview**
>
> Feel free to run this code block to visualize the data.
>
> ```
> #| echo: true
> df <- read.csv("candy-data.csv")
>
> barplot(table(df$fruity),
> names.arg = c("Not fruity","Fruity"),
> main = "Fruity indicator",
> ylab = "Count")
> ```

```
#| exercise: cat_q2_prop_fruity
#| exercise.lines: 6
#| echo: false
df <- read.csv("candy-data.csv")

mean(df$___ == ___)
```

Use the right indicator variable (0 or 1) for fruity candies.

Solution.

```
#| exercise: cat_q2_prop_fruity
#| solution: true
df <- read.csv("candy-data.csv")
mean(df$fruity == 1)
```

```
#| exercise: cat_q2_prop_fruity
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
exp <- mean(df$fruity == 1)

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

if (!ok) fail("Return a single numeric proportion.")
else if (abs(x - exp) < 1e-12) pass("Correct.")
else fail("Proportion mismatch. Check mean(fruity == 1).")
})
```

Q2 — Conditional Proportion

Return the proportion of hard candies among all fruity candies, i.e. return a single numeric value for P(Hard Candy = 1 | Fruity Candy = 1).

Photo by Joanna Kosinska on Unsplash

Preview

Feel free to run this code block to visualize the data.

```
#| echo: true
df <- read.csv("candy-data.csv")


tab <- table(df$fruity, df$hard)
mosaicplot(tab,
main = "Fruity vs Hard (mosaic plot)",
xlab = "Fruity (0/1)",
ylab = "Hard (0/1)")
```

```
#| exercise: cat_q3_cond_prop
#| exercise.lines: 8
#| echo: false
df <- read.csv("candy-data.csv")

sub <- subset(df, ___ == ___) #larger subset
mean(sub$___ == ___) #smaller subset
```

Subset to fruity == 1 and then compute mean(hard == 1) in that subset.

Solution.

```
#| exercise: cat_q3_cond_prop
#| solution: true
df <- read.csv("candy-data.csv")

sub <- subset(df, fruity == 1)
mean(sub$hard == 1)
```

```
#| exercise: cat_q3_cond_prop
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
exp <- mean(subset(df, fruity == 1)$hard == 1)

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

if (!ok) fail("Return a single numeric proportion.")
else if (abs(x - exp) < 1e-12) pass("Correct.")
else fail("Mismatch. Check the subset (fruity==1) and then mean(hard==1).")
})
```

Q3 — Difference in Conditional Proportions

Compute p_1 = P(bar == 1 | chocolate == 1), i.e. proportion of candy bars among all chocolate candies and then compute p_0 P(bar == 1 | chocolate == 0), i.e. proportion of candy bars among candies that are not made of chocolate. Finally return p_1 - p_0 .

Photo by Denny Müller on Unsplash

Preview

Feel free to run this code block to visualize the data.

```
#| echo: true
df <- read.csv("candy-data.csv")

mosaicplot(table(df$chocolate, df$bar),
main = "Chocolate vs Bar (mosaic plot)",
xlab = "Chocolate (0/1)",
ylab = "Bar (0/1)")
```

```
#| exercise: cat_q4_diff_props
#| exercise.lines: 12
#| echo: false
df <- read.csv("candy-data.csv")

p1 <- mean(subset(df, ___ == ___)$bar == ___)
p0 <- mean(subset(df, ___ == ___)$___ == ___)

p1 - p0
```

Use bar as the event and compare chocolate vs. no chocolate.

Solution.

```
#| exercise: cat_q4_diff_props
#| solution: true
df <- read.csv("candy-data.csv")

p1 <- mean(subset(df, chocolate == 1)$bar == 1)
p0 <- mean(subset(df, chocolate == 0)$bar == 1)

p1 - p0
```

```
#| exercise: cat_q4_diff_props
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
p1 <- mean(subset(df, chocolate == 1)$bar == 1)
p0 <- mean(subset(df, chocolate == 0)$bar == 1)
```

```
exp <- p1 - p0

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

if (!ok) fail("Return a single numeric difference p1 - p0.")
else if (abs(x - exp) < 1e-12) pass("Correct.")
else fail("Mismatch. Check your subsetting and mean(bar==1).")
})
```

Q4 — Chi-square Test of Independence

Perform a chi-sq test of independence between chocolate and bar and return the p-value.

Photo by Denny Müller on Unsplash

Info

A chi-sq test of independence essentially uses a contingency table to test: H_0: The variables are independent. vs. H_a: The variables are associated.
In R, we can test this by creating a table(x, y) of the two variables and then perform a test on them.

Preview

Feel free to run this code block to visualize the data.

```
#| echo: true
df <- read.csv("candy-data.csv")

tab <- table(df$chocolate, df$bar)
mosaicplot(tab,
main = "Chocolate vs Bar",
xlab = "Chocolate (0/1)",
ylab = "Bar (0/1)")
```

```
#| exercise: cat_q5_chisq_p
#| exercise.lines: 8
#| echo: false
df <- read.csv("candy-data.csv")
```

```
tab <- table(df$___, df$___)
___.test(tab, correct = FALSE)$___ #return p-value here
```

Make the appropriate table and fill in the correct test.

Solution.

```
#| exercise: cat_q5_chisq_p
#| solution: true
df <- read.csv("candy-data.csv")

tab <- table(df$chocolate, df$bar)
chisq.test(tab, correct = FALSE)$p.value
```

```
#| exercise: cat_q5_chisq_p
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
tab <- table(df$chocolate, df$bar)
exp <- chisq.test(tab, correct = FALSE)$p.value

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

if (!ok) fail("Return a single numeric p-value.")
else if (abs(x - exp) < 1e-12) pass("Correct chi-square p-value.")
else fail("Mismatch. Check table order and chisq.test(..., correct=FALSE).")
})
```

Q5 — Fisher's Exact Test

Run Fisher's exact test for association between fruity and hard candies. Return the p-value.

Photo by Joanna Kosinska on Unsplash

> **Info**
>
> Fisher's exact test is often used for 2×2 tables, especially when some expected counts may be small.It's a non-parametric test, meaning it doesn't assume data follows a specific distribution, making it more accurate than the Chi-squared test when expected counts are low.

Preview

Feel free to run this code block to visualize the data.

```
#| echo: true
df <- read.csv("candy-data.csv")

tab <- table(df$fruity, df$hard)
mosaicplot(tab,
main = "Fruity vs Hard",
xlab = "Fruity (0/1)",
ylab = "Hard (0/1)")
```

```
#| exercise: cat_q6_fisher_p
#| exercise.lines: 8
#| echo: false
df <- read.csv("candy-data.csv")

tab <- table(df$___, df$___)
___.test(tab)$___
```

Make the appropriate table and fill in the correct test.

Solution.

```
#| exercise: cat_q6_fisher_p
#| solution: true
df <- read.csv("candy-data.csv")

tab <- table(df$fruity, df$hard)
fisher.test(tab)$p.value
```

```
#| exercise: cat_q6_fisher_p
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
tab <- table(df$fruity, df$hard)
exp <- fisher.test(tab)$p.value

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)
```

```
if (!ok) fail("Return a single numeric p-value.")
else if (abs(x - exp) < 1e-12) pass("Correct Fisher p-value.")
else fail("Mismatch. Check your table variables and fisher.test(tab).")
})
```

Q6 — Create a Categorical Outcome

Data for this dataset was collected from actual people by creating a website where participants were presented with two fun-sized candies and asked to click on the one they would prefer to receive. In total, more than 269 thousand votes were collected from 8,371 different IP addresses.

Hence, each candy has a win percent rate associated with it. For this question:

Create high_win as: “High” if winpercent >= median(winpercent) “Low” otherwise

Then test whether the proportion of High differs between chocolate and non-chocolate candies using prop.test(…). Return the p-value.

> **Info**
>
> This is a two-proportion test comparing:
> group 1: chocolate = 1
> group 2: chocolate = 0 for the “success” outcome: high_win == “High”

> **Preview**
>
> Feel free to run this code block to visualize the data.

```
#| echo: true
df <- read.csv("candy-data.csv")

cutoff <- median(df$winpercent)
high_win <- ifelse(df$winpercent >= cutoff, "High", "Low")

# Visual only: stacked bar chart

barplot(table(df$chocolate, high_win),
beside = FALSE,
legend.text = c("Chocolate=0","Chocolate=1"),
main = "High/Low win by Chocolate",
xlab = "High win group",
ylab = "Count")
```

```
#| exercise: cat_q7_prop_test
#| exercise.lines: 18
#| echo: false
df <- read.csv("candy-data.csv")

cutoff <- ___(df$winpercent)
high_win <- ifelse(df$winpercent >= ___, "High", "Low")

x1 <- sum(high_win[df$chocolate == ___] == "High")
n1 <- sum(df$chocolate == ___)

x2 <- sum(high_win[df$chocolate == ___] == "High")
n2 <- sum(df$chocolate == ___)

___.test(c(x1, x2), c(n1, n2), correct = FALSE)$___
```

Find the cutoff by computing the median and then add in the appropriate indicator variable in the blanks.

Solution.

```
#| exercise: cat_q7_prop_test
#| solution: true
df <- read.csv("candy-data.csv")

cutoff <- median(df$winpercent)
```

```
high_win <- ifelse(df$winpercent >= cutoff, "High", "Low")

x1 <- sum(high_win[df$chocolate == 1] == "High")
n1 <- sum(df$chocolate == 1)

x2 <- sum(high_win[df$chocolate == 0] == "High")
n2 <- sum(df$chocolate == 0)

prop.test(c(x1, x2), c(n1, n2), correct = FALSE)$p.value
```

```
#| exercise: cat_q7_prop_test
#| check: true
gradethis::grade_this({
df <- read.csv("candy-data.csv")
cutoff <- median(df$winpercent)
high_win <- ifelse(df$winpercent >= cutoff, "High", "Low")

x1 <- sum(high_win[df$chocolate == 1] == "High")
n1 <- sum(df$chocolate == 1)

x2 <- sum(high_win[df$chocolate == 0] == "High")
n2 <- sum(df$chocolate == 0)

exp <- prop.test(c(x1, x2), c(n1, n2), correct = FALSE)$p.value

x <- .result
ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

if (!ok) fail("Return a single numeric p-value.")
else if (abs(x - exp) < 1e-12) pass("Correct two-proportion test p-value.")
else fail("Mismatch. Check x1,n1,x2,n2 and prop.test(..., correct=FALSE).")
})
```

Q7 — Odds Ratio

Compute the odds ratio for the 2×2 table of chocolate (rows) vs bar (columns)

$$OR = (a/b)/(c/d)$$

where: - a = #(chocolate=1, bar=1) - b = #(chocolate=1, bar=0) - c = #(chocolate=0, bar=1) - d = #(chocolate=0, bar=0)

Photo by Denny Müller on Unsplash

Info

Odds ratio is a common association measure for 2×2 categorical data:
OR $>$ 1 suggests positive association
OR $=$ 1 suggests no association
OR $<$ 1 suggests negative association

Preview

```
#| echo: true
df <- read.csv("candy-data.csv")
mosaicplot(table(df$chocolate, df$bar),
main = "Chocolate vs Bar",
xlab = "Chocolate (0/1)",
ylab = "Bar (0/1)")
```

```
#| exercise: cat_q8_odds_ratio
#| exercise.lines: 16
#| echo: false
df <- read.csv("candy-data.csv")

tab <- table(df$___, df$___)

a <- tab["1","1"]
b <- tab["1","0"]
c <- tab["0","1"]
d <- tab["0","0"]

odds_choc1 <- __ / __
odds_choc0 <- __ / __

odds_ratio <- odds_choc1 / odds_choc0
```

Use the Odd Ratio formula after making the appropriate table.

Solution.

```
#| exercise: cat_q8_odds_ratio
#| solution: true
df <- read.csv("candy-data.csv")
```

```
tab <- table(df$chocolate, df$bar)

a <- tab["1","1"]
b <- tab["1","0"]
c <- tab["0","1"]
d <- tab["0","0"]

odds_choc1 <- a / b
odds_choc0 <- c / d

(or <- odds_choc1 / odds_choc0)
```

```
#| exercise: cat_q8_odds_ratio
#| check: true
gradethis::grade_this({
  df <- tryCatch(read.csv("candy-data.csv"), error = function(e) NULL)
  if (is.null(df)) fail("Couldn't read 'candy-data.csv'.")

  if (!all(c("chocolate","bar") %in% names(df))) {
    fail("CSV must contain 'chocolate' and 'bar' columns.")
  }

  tab <- table(df$chocolate, df$bar)

  a <- tab["1","1"]
  b <- tab["1","0"]
  c <- tab["0","1"]
  d <- tab["0","0"]

  exp <- as.numeric((a * d) / (b * c))

  x <- .result
  ok <- is.numeric(x) && length(x) == 1L && is.finite(x)

  if (!ok) fail("Return a single numeric odds ratio.")
  else if (isTRUE(all.equal(as.numeric(x), exp, tolerance = 1e-12))) pass("Correct odds ratio
  else fail("Mismatch. Check the odds (a/b) and (c/d), then take their ratio.")
})
```