6. Mehrfachantworten auswerten

Multiple-Choice-Fragen mit mehreren möglichen Antworten

Dr. Paul Schmidt

Was sind Mehrfachantworten?

Bei vielen Umfragen und Interviews gibt es Fragen, bei denen Befragte nicht nur eine, sondern mehrere Antwortmöglichkeiten auswählen können. Ein klassisches Beispiel wäre die Frage “Welche dieser Obstsorten mögen Sie?” mit fünf Auswahlmöglichkeiten. Anders als bei einer Single-Choice-Frage, wo genau eine Option gewählt werden muss, kann hier jede Person beliebig viele Optionen ankreuzen — von keiner einzigen bis hin zu allen fünf.

Diese scheinbar simple Erweiterung hat weitreichende Konsequenzen für die Datenstruktur und Auswertung: Während bei Single-Choice eine Spalte mit den gewählten Kategorien ausreicht, brauchen wir bei Multiple-Choice andere Ansätze. Stellen wir uns vor, vier Personen beantworten unsere Obstfrage:

Welche dieser Obstsorten mögen Sie?

Person 1

- ☑ Apfel
- ☐ Banane
- ☑ Kirsche
- ☐ Mango
- ☑ Orange

Person 2

- ☐ Apfel
- ☑ Banane
- ☐ Kirsche
- ☐ Mango
- ☐ Orange

Person 3

- ☑ Apfel
- ☑ Banane
- ☑ Kirsche
- ☑ Mango
- ☑ Orange

Person 4

- ☑ Apfel
- ☐ Banane
- ☑ Kirsche
- ☑ Mango
- ☐ Orange

Person 1 mag drei Obstsorten, Person 2 nur eine einzige, Person 3 alle fünf und Person 4 wieder drei. Wie speichern wir diese Informationen in einer Tabelle? Dafür gibt es verschiedene Formate.

Datenformate für Mehrfachantworten

Dichotomes Format (Wide)

Das häufigste Format in der Praxis: Jede Antwortoption wird zu einer eigenen Spalte mit den Werten 0 (nicht gewählt) und 1 (gewählt). Die Tabelle wird dadurch “breit”, weshalb man auch vom Wide-Format spricht.

person_id	Q1_apfel	Q1_banane	Q1_kirsche	Q1_mango	Q1_orange
1	1	0	1	1	0
2	0	0	0	0	0
3	1	1	1	1	1
4	1	0	1	1	0

Dieses Format ist typisch für Exporte aus Umfragetools wie Google Forms, SurveyMonkey, Qualtrics, LimeSurvey oder REDCap. Die zusammengehörigen Spalten einer Mehrfachantwort-Frage haben dabei oft einen gemeinsamen Präfix (hier: `Q1_`), was das Erkennen und Auswählen der Spalten erleichtert. Der Vorteil: Jede Zeile ist eine Person, und man kann sofort sehen, welche Kombinationen gewählt wurden. Der Nachteil: Bei vielen Antwortmöglichkeiten wird die Tabelle sehr breit.

Collapsed Format (Delimited)

Eine kompaktere Alternative: Alle gewählten Optionen stehen in einer einzigen Spalte, getrennt durch ein Trennzeichen wie Semikolon, Komma oder Pipe.

person_id	Q1_obst
1	Apfel; Kirsche; Mango
2	
3	Apfel; Banane; Kirsche; Mango; Orange
4	Apfel; Kirsche; Mango

Dieses Format entsteht oft bei manueller Dateneingabe in Excel oder bei älteren Datenbanksystemen. Es ist platzsparend und für Menschen gut lesbar, aber für die statistische Auswertung muss es erst in ein anderes Format überführt werden.

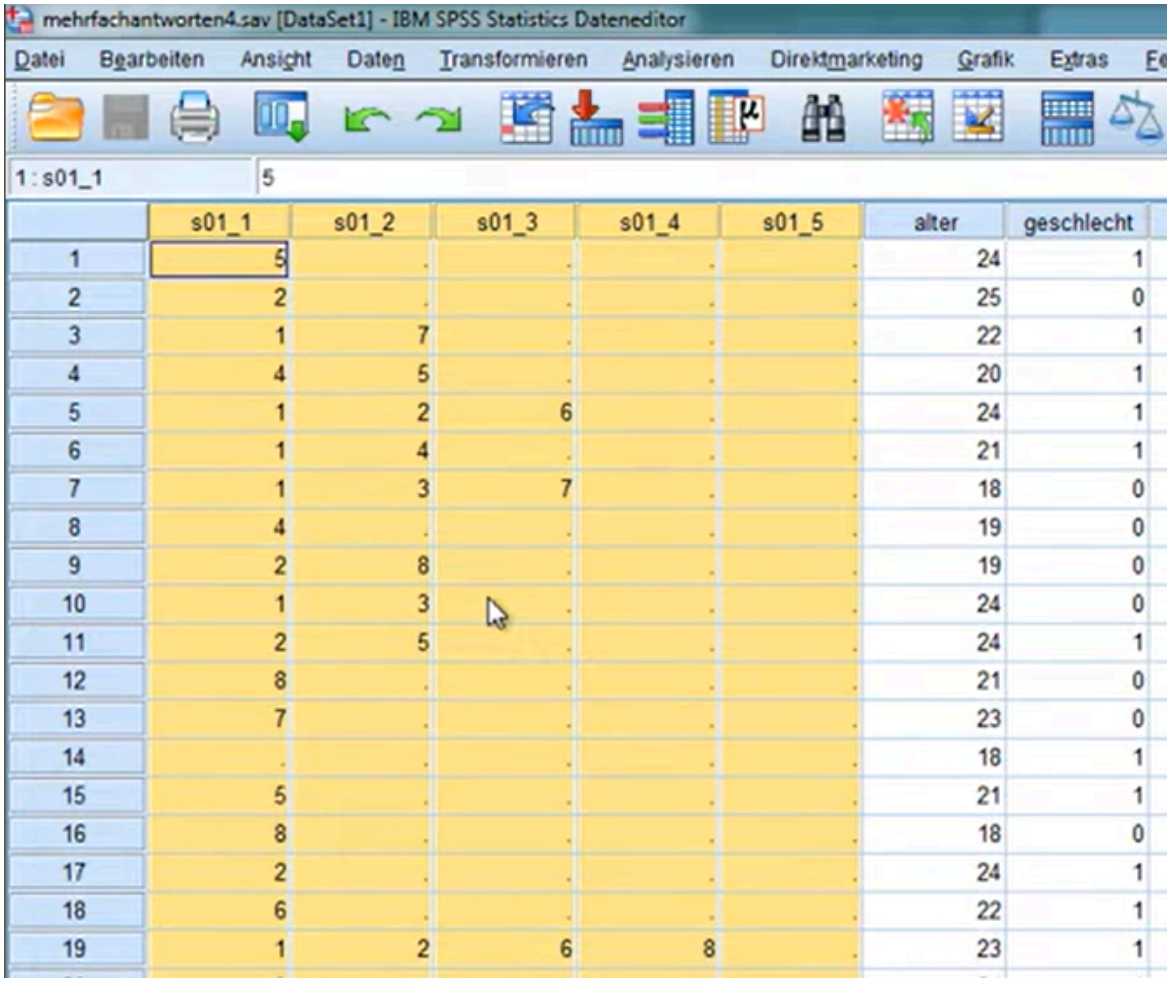
Long Format

Im Long-Format gibt es eine Zeile pro gewählter Option. Das bedeutet: Eine Person, die drei Obstsorten gewählt hat, erscheint in drei Zeilen.

person_id	obst
1	Apfel
1	Kirsche
1	Mango
2	
3	Apfel
3	Banane
3	Kirsche
3	Mango
3	Orange
4	Apfel
4	Kirsche
4	Mango

Das Long-Format ist typisch für relationale Datenbanken und besonders gut geeignet für Auswertungen mit tidyverse-Funktionen wie `count()` oder `group_by()`. Der Nachteil: Die Tabelle wird bei vielen Antworten sehr lang, und Personen, die nichts gewählt haben, erscheinen gar nicht.

> ℹ Für SPSS-Umsteiger: Multiple Response Sets
>
> In SPSS müssen Mehrfachantworten über einen speziellen Mechanismus ausgewertet werden: Man definiert zunächst ein “Multiple Response Set” aus den einzelnen Variablen, bevor man Häufigkeiten berechnen kann. Dabei unterscheidet SPSS zwischen der Auszählung “nach Fällen” (Prozent der Befragten) und “nach Antworten” (Prozent aller gegebenen Antworten).



	s01_1	s01_2	s01_3	s01_4	s01_5	alter	geschlecht
1	5	.	.	.	.	24	1
2	2	.	.	.	.	25	0
3	1	7	.	.	.	22	1
4	4	5	.	.	.	20	1
5	1	2	6	.	.	24	1
6	1	4	.	.	.	21	1
7	1	3	7	.	.	18	0
8	4	.	.	.	.	19	0
9	2	8	.	.	.	19	0
10	1	3	.	.	.	24	0
11	2	5	.	.	.	24	1
12	8	.	.	.	.	21	0
13	7	.	.	.	.	23	0
14	.	.	.	.	.	18	1
15	5	.	.	.	.	21	1
16	8	.	.	.	.	18	0
17	2	.	.	.	.	24	1
18	6	.	.	.	.	22	1
19	1	2	6	8	.	23	1

Abbildung 1: SPSS Dateneditor mit Mehrfachantworten-Variablen

In R ist dieser Umweg über Set-Definitionen nicht nötig — wir arbeiten direkt mit den Daten in einem der drei Formate. Wer dennoch eine SPSS-ähnliche Syntax bevorzugt, kann die R-Pakete `expss` (mit `mrset()` und `tab_stat_cpct()`) oder `sjmisc` (mit `frq()` und `flat_table()`) nutzen.

Eine ausführliche Anleitung zur SPSS-Vorgehensweise zeigt dieses Video: Mehrfachantworten in SPSS auswerten

Zwischen Formaten konvertieren

In der Praxis erhält man Daten oft in einem Format, möchte sie aber in einem anderen auswerten. Mit tidyverse-Funktionen lässt sich jedes Format in jedes andere überführen. Wir definieren zunächst unsere drei Beispieldatensätze explizit:

```r
# Dichotomes Format (Wide)
obst_dichotom <- tibble(
  person_id = 1:4,
  Q1_apfel   = c(1, 0, 1, 1),
  Q1_banane  = c(0, 1, 1, 0),
  Q1_kirsche = c(1, 0, 1, 1),
  Q1_mango   = c(0, 0, 1, 1),
  Q1_orange  = c(1, 0, 1, 0)
)

# Collapsed Format
obst_collapsed <- tibble(
  person_id = 1:4,
  Q1_obst = c("Apfel; Kirsche; Orange",
              "Banane",
              "Apfel; Banane; Kirsche; Mango; Orange",
              "Apfel; Kirsche; Mango")
)

# Long Format
obst_long <- tibble(
  person_id = c(1, 1, 1, 2, 3, 3, 3, 3, 3, 4, 4, 4),
  obst = c("Apfel", "Kirsche", "Orange",
           "Banane",
           "Apfel", "Banane", "Kirsche", "Mango", "Orange",
           "Apfel", "Kirsche", "Mango")
)
```

Dichotom → Long

Das dichotome Format wird mit `pivot_longer()` ins Long-Format überführt. Der gemeinsame Präfix `Q1_` ermöglicht es, die zusammengehörigen Spalten mit `starts_with()` auszuwählen. Anschließend filtern wir nur die Zeilen, in denen eine Option gewählt wurde (Wert = 1).

```r
obst_dichotom %>%
  pivot_longer(
    cols = starts_with("Q1_"),        # alle Spalten mit Präfix Q1_
    names_to = "obst",                # Spaltennamen werden zu Werten
    names_prefix = "Q1_",             # Präfix entfernen
    values_to = "gewaehlt"            # 0/1-Werte in neue Spalte
  ) %>%
  filter(gewaehlt == 1) %>%           # nur gewählte Optionen behalten
  select(-gewaehlt)                   # Hilfsspalte entfernen
```

```
# A tibble: 12 × 2
   person_id obst
       <int> <chr>
 1         1 apfel
 2         1 kirsche
 3         1 orange
 4         2 banane
 5         3 apfel
 6         3 banane
 7         3 kirsche
 8         3 mango
 9         3 orange
10         4 apfel
11         4 kirsche
12         4 mango
```

Dichotom → Collapsed

Für die Konvertierung ins Collapsed-Format pivotieren wir zunächst ins Long-Format, filtern die gewählten Optionen und fassen sie dann mit `summarise()` und `str_c()` zusammen.

```
obst_dichotom %>%
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  summarise(
    Q1_obst = str_c(str_to_title(obst), collapse = "; "),
    .by = person_id
  )
```

```
# A tibble: 4 × 2
  person_id Q1_obst
      <int> <chr>
1         1 Apfel; Kirsche; Orange
2         2 Banane
3         3 Apfel; Banane; Kirsche; Mango; Orange
4         4 Apfel; Kirsche; Mango
```

Long → Dichotom

Der umgekehrte Weg: Wir fügen eine Hilfsspalte mit dem Wert 1 hinzu und pivotieren dann mit `pivot_wider()` ins breite Format. Fehlende Werte werden mit 0 aufgefüllt.

```
obst_long %>%
  mutate(
    obst = str_to_lower(obst),        # einheitliche Schreibweise
    gewaehlt = 1                      # Hilfsspalte
  ) %>%
  pivot_wider(
    names_from = obst,
    names_prefix = "Q1_",             # Präfix hinzufügen
    values_from = gewaehlt,
    values_fill = 0                   # nicht gewählte = 0
  )
```

```
# A tibble: 4 × 6
  person_id Q1_apfel Q1_kirsche Q1_orange Q1_banane Q1_mango
      <dbl>    <dbl>      <dbl>     <dbl>     <dbl>    <dbl>
1         1        1          1         1         0        0
2         2        0          0         0         1        0
3         3        1          1         1         1        1
4         4        1          1         0         0        1
```

Long → Collapsed

Im Long-Format gruppieren wir nach Person und fügen die Obstsorten zu einem String zusammen.

```
obst_long %>%
  summarise(
    Q1_obst = str_c(obst, collapse = "; "),
    .by = person_id
  )
```

```
# A tibble: 4 × 2
  person_id Q1_obst
      <dbl> <chr>
1         1 Apfel; Kirsche; Orange
2         2 Banane
3         3 Apfel; Banane; Kirsche; Mango; Orange
4         4 Apfel; Kirsche; Mango
```

Collapsed → Long

Das Collapsed-Format wird mit `separate_longer_delim()` aufgetrennt. Diese Funktion erzeugt für jeden Wert zwischen den Trennzeichen eine eigene Zeile.

```
obst_collapsed %>%
  separate_longer_delim(Q1_obst, delim = "; ") %>%
  rename(obst = Q1_obst) %>%          # Spalte umbenennen
  mutate(obst = str_trim(obst))       # eventuelle Leerzeichen entfernen
```

```
# A tibble: 12 × 2
   person_id obst
       <int> <chr>
 1         1 Apfel
 2         1 Kirsche
 3         1 Orange
 4         2 Banane
 5         3 Apfel
 6         3 Banane
 7         3 Kirsche
 8         3 Mango
 9         3 Orange
10         4 Apfel
11         4 Kirsche
12         4 Mango
```

Collapsed → Dichotom

Für die Konvertierung ins dichotome Format gehen wir den Umweg über das Long-Format.

```
obst_collapsed %>%
  separate_longer_delim(Q1_obst, delim = "; ") %>%
  mutate(
    obst = str_to_lower(str_trim(Q1_obst)),
    gewaehlt = 1
  ) %>%
  select(-Q1_obst) %>%
  pivot_wider(
    names_from = obst,
    names_prefix = "Q1_",
    values_from = gewaehlt,
    values_fill = 0
  )
```

```
# A tibble: 4 × 6
  person_id Q1_apfel Q1_kirsche Q1_orange Q1_banane Q1_mango
      <int>    <dbl>      <dbl>     <dbl>     <dbl>    <dbl>
1         1        1          1         1         0        0
2         2        0          0         0         1        0
3         3        1          1         1         1        1
4         4        1          1         0         0        1
```

Auswertung: Häufigkeiten

Für die bisherigen Beispiele haben wir einen minimalen Datensatz mit nur vier Personen verwendet, um die Formate übersichtlich zu demonstrieren. Für eine realistische Auswertung arbeiten wir nun mit einem größeren Datensatz: 20 Befragte (9 männlich, 11 weiblich) haben dieselbe Obstfrage beantwortet.

```
# Größerer Datensatz im dichotomen Format
umfrage <- tibble(
  person_id = 1:20,
  geschlecht = c("m", "w", "w", "m", "w", "m", "w", "w", "m", "w",
                 "w", "m", "w", "m", "w", "m", "w", "w", "m", "w"),
  Q1_apfel   = c(1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1),
  Q1_banane  = c(0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0),
  Q1_kirsche = c(1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1),
  Q1_mango   = c(0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1),
  Q1_orange  = c(1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0)
)

umfrage
```

```
# A tibble: 20 × 7
   person_id geschlecht Q1_apfel Q1_banane Q1_kirsche Q1_mango Q1_orange
       <int> <chr>         <dbl>     <dbl>      <dbl>    <dbl>     <dbl>
 1         1 m                 1         0          1        0         1
 2         2 w                 1         1          1        0         0
 3         3 w                 0         1          1        1         0
 4         4 m                 1         0          0        1         1
 5         5 w                 1         1          1        0         0
 6         6 m                 0         1          0        0         1
 7         7 w                 1         1          1        1         1
 8         8 w                 1         1          0        0         1
 9         9 m                 1         0          1        0         1
10        10 w                 0         1          1        1         0
11        11 w                 1         1          1        0         0
12        12 m                 1         0          0        1         1
13        13 w                 0         1          1        0         0
14        14 m                 1         0          1        0         1
15        15 w                 1         1          0        1         0
16        16 m                 0         1          1        0         0
17        17 w                 1         0          1        1         0
18        18 w                 1         1          0        0         1
19        19 m                 0         1          1        0         0
20        20 w                 1         0          1        1         0
```

Übung: Formatkonvertierung

Nutzen Sie das Gelernte, um den `umfrage`-Datensatz in die anderen beiden Formate zu konvertieren.

Aufgabe A: Konvertieren Sie `umfrage` ins Long-Format und ins Collapsed-Format. Die Spalte `geschlecht` können Sie dabei zunächst ignorieren (einfach weglassen).

Aufgabe B (Bonus): Konvertieren Sie `umfrage` so ins Long-Format, dass die Spalte `geschlecht` erhalten bleibt und korrekt jeder Zeile zugeordnet ist.

Lösungsvorschlag

Aufgabe A: Long-Format (ohne Geschlecht)

```
umfrage %>%
  select(-geschlecht) %>%            # Geschlecht weglassen
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  select(-gewaehlt)
```

```
# A tibble: 58 × 2
   person_id obst
       <int> <chr>
 1         1 apfel
 2         1 kirsche
 3         1 orange
 4         2 apfel
 5         2 banane
 6         2 kirsche
 7         3 banane
 8         3 kirsche
 9         3 mango
10         4 apfel
# i 48 more rows
```

Aufgabe A: Collapsed-Format (ohne Geschlecht)

```
umfrage %>%
  select(-geschlecht) %>%
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  summarise(
    Q1_obst = str_c(str_to_title(obst), collapse = "; "),
    .by = person_id
  )
```

```
# A tibble: 20 × 2
   person_id Q1_obst
       <int> <chr>
 1         1 Apfel; Kirsche; Orange
 2         2 Apfel; Banane; Kirsche
 3         3 Banane; Kirsche; Mango
 4         4 Apfel; Mango; Orange
 5         5 Apfel; Banane; Kirsche
 6         6 Banane; Orange
 7         7 Apfel; Banane; Kirsche; Mango; Orange
 8         8 Apfel; Banane; Orange
 9         9 Apfel; Kirsche; Orange
10        10 Banane; Kirsche; Mango
11        11 Apfel; Banane; Kirsche
12        12 Apfel; Mango; Orange
13        13 Banane; Kirsche
14        14 Apfel; Kirsche; Orange
15        15 Apfel; Banane; Mango
16        16 Banane; Kirsche
```

```
# A tibble: 58 × 3
   person_id geschlecht obst
       <int> <chr>     <chr>
 1         1 m         apfel
 2         1 m         kirsche
umfrage %>%
pivot_longer(
  cols = starts_with("Q1_"),       # nur Q1 -Spalten pivotieren
```

Der Trick bei Aufgabe B: Mit `starts_with("Q1_")` werden nur die Mehrfachantwort-Spalten pivotiert, `geschlecht` bleibt automatisch erhalten.

Häufigkeitstabellen mit tabyl()

Sobald die Daten im Long-Format vorliegen, können wir die vertrauten Werkzeuge aus dem Kapitel zu Häufigkeitstabellen nutzen. Zunächst bringen wir den Datensatz ins Long-Format:

```
umfrage_long <- umfrage %>%
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  select(-gewaehlt)
```

Jetzt können wir mit `tabyl()` aus dem janitor-Paket eine Häufigkeitstabelle erstellen — genau wie bei jeder anderen kategorialen Variable:

```
umfrage_long %>%
  tabyl(obst) %>%
  adorn_pct_formatting()
```

```
    obst  n percent
   apfel 14   24.1%
  banane 13   22.4%
 kirsche 14   24.1%
   mango  8   13.8%
  orange  9   15.5%
```

Prozent der Fälle vs. Prozent der Antworten

Bei Mehrfachantworten gibt es eine wichtige Unterscheidung, die `tabyl()` nicht automatisch macht: Die Prozentangaben beziehen sich auf die Anzahl der *Antworten* (hier: 56), nicht auf die Anzahl der *Befragten* (hier: 20).

Prozent der Antworten (was `tabyl()` liefert) beantwortet die Frage: “Welchen Anteil macht diese Option an *allen gegebenen Antworten* aus?” Diese Prozente summieren sich auf genau 100%.

Prozent der Fälle beantwortet die Frage: “Wie viel Prozent der *Befragten* haben diese Option gewählt?” Da jede Person mehrere Optionen wählen kann, summieren sich diese Prozente auf mehr als 100%.

```
n_personen <- n_distinct(umfrage$person_id)

umfrage_long %>%
  count(obst, name = "n") %>%
  mutate(
    pct_antworten = n / sum(n) * 100,
    pct_faelle = n / n_personen * 100
  ) %>%
  mutate(across(starts_with("pct"), \(x) round(x, 1)))
```

```
# A tibble: 5 × 4
  obst        n pct_antworten pct_faelle
  <chr>   <int>         <dbl>      <dbl>
1 apfel      14          24.1         70
2 banane     13          22.4         65
3 kirsche    14          24.1         70
```

```
4 mango       8          13.8         40
5 orange      9          15.5         45
```

! Achtung: Fehlende Werte vs. “nichts gewählt”

Bei der Berechnung von “Prozent der Fälle” ist die Frage: Was ist die Basis?

- **Alle Spalten = 0:** Die Person hat die Frage gesehen und aktiv nichts gewählt
- **Alle Spalten = NA:** Die Person hat die Frage übersprungen (fehlende Werte)

Im Long-Format verschwinden beide Fälle — es gibt keine Zeile, wenn nichts gewählt wurde. Vor der Auswertung sollte man daher prüfen, ob solche Fälle existieren:

```
# Personen ohne Antwort (alle 0)
umfrage %>%
  filter(rowSums(across(starts_with("Q1_"))) == 0)

# Personen mit fehlenden Werten
umfrage %>%
  filter(if_any(starts_with("Q1_"), is.na))
```

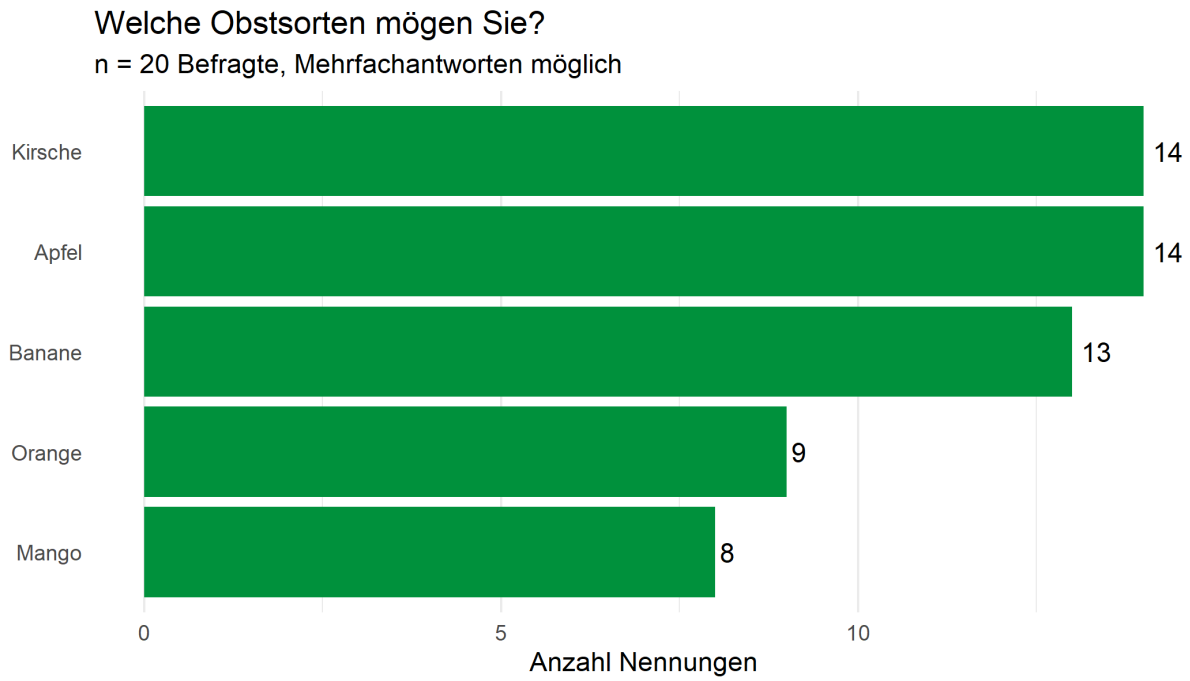
Je nach Fragestellung zählen diese Personen zur Basis (n) oder nicht.

In der Praxis ist **Prozent der Fälle** meist aussagekräftiger, weil man so direkt sagen kann: “65% der Befragten mögen Äpfel.” Das Prozent der Antworten ist eher relevant, wenn man die relative Beliebtheit der Optionen untereinander vergleichen möchte.

Visualisierung

Ein einfaches Balkendiagramm zeigt die Häufigkeiten auf einen Blick:

```
umfrage_long %>%
  count(obst) %>%
  mutate(obst = str_to_title(obst)) %>%
  ggplot(aes(x = reorder(obst, n), y = n)) +
  geom_col(fill = "#00923f") +
  geom_text(aes(label = n), hjust = -0.3, size = 4) +
  coord_flip() +
  labs(
    x = NULL,
    y = "Anzahl Nennungen",
    title = "Welche Obstsorten mögen Sie?",
    subtitle = glue::glue("n = {n_personen} Befragte, Mehrfachantworten möglich")
  ) +
  theme_minimal() +
  theme(panel.grid.major.y = element_blank())
```



Auswertung: Kreuztabellen

Häufig möchte man wissen, ob sich die Antworten zwischen Gruppen unterscheiden. Mit unserer Gruppierungsvariable `geschlecht` können wir Kreuztabellen erstellen — wieder mit den vertrauten Werkzeugen:

```
umfrage_long %>%
  tabyl(obst, geschlecht) %>%
  adorn_totals("col") %>%
  adorn_percentages("col") %>%
  adorn_pct_formatting() %>%
  adorn_ns()
```

```
    obst          m           w       Total
   apfel 23.8% (5) 24.3%  (9) 24.1% (14)
  banane 14.3% (3) 27.0% (10) 22.4% (13)
 kirsche 23.8% (5) 24.3%  (9) 24.1% (14)
   mango  9.5% (2) 16.2%  (6) 13.8%  (8)
  orange 28.6% (6)  8.1%  (3) 15.5%  (9)
```

Diese Tabelle zeigt für jedes Geschlecht, welchen Anteil die jeweilige Obstsorte an allen Antworten dieser Gruppe ausmacht. Beachte: Auch hier handelt es sich um “Prozent der Antworten”, nicht “Prozent der Fälle”.

Kombinationsmuster (fortgeschritten)

Eine weitere interessante Frage: Welche Obstsorten werden häufig zusammen gewählt? Diese Analyse geht über einfache Häufigkeiten hinaus und untersucht die Muster in den Antworten.

Die häufigsten Kombinationen

Dafür nutzen wir den gleichen Ansatz wie bei der Formatkonvertierung — wir pivotieren ins Long-Format und fassen die gewählten Optionen pro Person zusammen:

```
umfrage %>%
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  summarise(
    kombination = str_c(str_to_title(obst), collapse = " + "),
    .by = person_id
  ) %>%
  count(kombination, sort = TRUE, name = "anzahl")
```

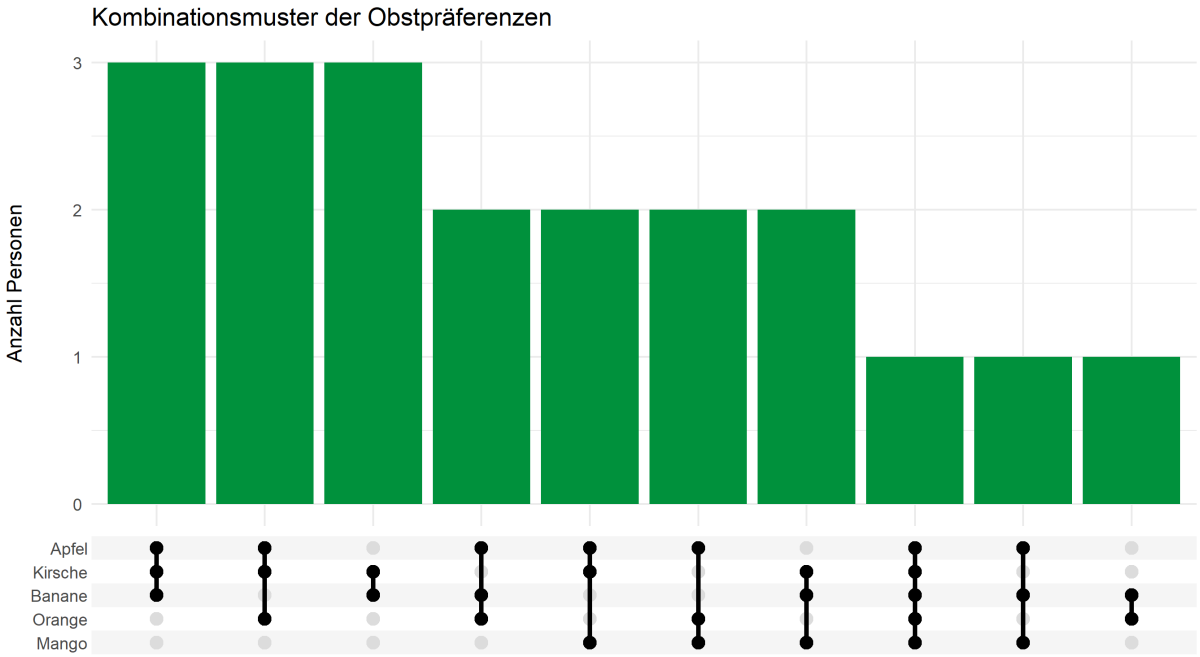
```
# A tibble: 10 × 2
   kombination                                anzahl
   <chr>                                       <int>
 1 Apfel + Banane + Kirsche                        3
 2 Apfel + Kirsche + Orange                        3
 3 Banane + Kirsche                                3
 4 Apfel + Banane + Orange                         2
 5 Apfel + Kirsche + Mango                         2
 6 Apfel + Mango + Orange                          2
 7 Banane + Kirsche + Mango                        2
 8 Apfel + Banane + Kirsche + Mango + Orange       1
 9 Apfel + Banane + Mango                          1
10 Banane + Orange                                 1
```

Übersicht mit UpSet-Plot

Für die Visualisierung von Kombinationen eignen sich UpSet-Plots besser als klassische Venn-Diagramme, besonders wenn es mehr als drei Kategorien gibt. Das Paket `ggupset` bietet eine ggplot2-Integration:

```
umfrage_long %>%
  summarise(
    obst = list(str_to_title(obst)),
    .by = person_id
  ) %>%
  ggplot(aes(x = obst)) +
  geom_bar(fill = "#00923f") +
  scale_x_upset() +
  labs(
    x = NULL,
    y = "Anzahl Personen",
    title = "Kombinationsmuster der Obstpräferenzen"
  ) +
  theme_minimal()
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
i The deprecated feature was likely used in the ggupset package.
  Please report the issue at <https://github.com/const-ae/ggupset/issues>.
```



Übung: Eigene Auswertung

Analysieren Sie den `umfrage`-Datensatz weiter:

1. Berechnen Sie die durchschnittliche Anzahl gewählter Obstsorten pro Person.
2. Gibt es einen Unterschied zwischen Männern und Frauen in der Anzahl der gewählten Optionen?
3. Welche Obstsorte wird am häufigsten als *einzige* Option gewählt (also von Personen, die nur eine Sorte mögen)?

ℹ Lösungsvorschlag

1. Durchschnittliche Anzahl pro Person

```
umfrage %>%
  mutate(
    anzahl_gewaehlt = rowSums(across(starts_with("Q1_")))
  ) %>%
  summarise(
    mittelwert = mean(anzahl_gewaehlt),
    median = median(anzahl_gewaehlt),
    min = min(anzahl_gewaehlt),
    max = max(anzahl_gewaehlt)
  )
```

```
# A tibble: 1 × 4
  mittelwert median   min   max
       <dbl>  <dbl> <dbl> <dbl>
1        2.9      3     2     5
```

2. Unterschied nach Geschlecht

```
umfrage %>%
  mutate(
    anzahl_gewaehlt = rowSums(across(starts_with("Q1_")))
  ) %>%
  summarise(
    mittelwert = mean(anzahl_gewaehlt),
    .by = geschlecht
  )
```

```
# A tibble: 2 × 2
  geschlecht mittelwert
  <chr>           <dbl>
1 m                2.62
2 w                3.08
```

3. Häufigste Einzelwahl

```
umfrage %>%
  mutate(
    anzahl_gewaehlt = rowSums(across(starts_with("Q1_")))
  ) %>%
  filter(anzahl_gewaehlt == 1) %>%        # nur Personen mit einer Wahl
  pivot_longer(
    cols = starts_with("Q1_"),
    names_to = "obst",
    names_prefix = "Q1_",
    values_to = "gewaehlt"
  ) %>%
  filter(gewaehlt == 1) %>%
  count(obst, sort = TRUE)
```

```
# A tibble: 0 × 2
# i 2 variables: obst <chr>, n <int>
```

Zusammenfassung

Mehrfachantworten erfordern besondere Aufmerksamkeit bei der Datenstruktur und Auswertung:

- **Drei gängige Formate:** Dichotom (0/1-Spalten), Collapsed (Trennzeichen), Long (eine Zeile pro Antwort)
- **Spaltenpräfixe:** Zusammengehörige Spalten haben oft einen gemeinsamen Präfix (z.B. `Q1_`), der mit `starts_with()` ausgewählt werden kann
- **Konvertierung:** Mit `pivot_longer()`, `pivot_wider()`, `separate_longer_delim()` und `summarise()` lassen sich alle Formate ineinander überführen
- **Häufigkeitstabellen:** Im Long-Format funktioniert `tabyl()` wie gewohnt
- **Zwei Prozentuierungsarten:** Prozent der Fälle (Basis: Personen) vs. Prozent der Antworten (Basis: alle Antworten)
- **Kreuztabellen:** Gruppierungsvariablen ermöglichen Vergleiche zwischen Subgruppen
- **Kombinationsmuster:** Zeigen, welche Optionen häufig zusammen gewählt werden (UpSet-Plot)

Bibliography