MoT Thesis Marc van der Toorn

2025-09-15

Regarding the dataset

Variables

Field name	Data type	Description
age	Integer	Age in years
education.level	Ordinal	Level of education: “Don’t know / not applicable”, ‘No formal qualifications’, ‘Prefer not to say’, ‘Secondary education (e.g. GED/GCSE)’, ‘High school diploma / A-levels’, ‘Technical/community college’, ‘Undergraduate degree (BA/BSc/other)’,‘Graduate degree (MA/MSc/MPhil/other)’, ‘Doctorate degree (PhD/other)’
household.income	Ordinal	Annual household income range: ‘Less than $60,000’, ‘$60,000 - $79,999’, ‘$80,000 - $99,999’, ‘$100,000 - $119,999’, ‘$120,000 - $139,999’, ‘$140,000 - $159,999’, ‘$160,000 - $179,999’, ‘More than $180,000’, ‘Prefer not to say’
physical.act	Ordinal	Physical activity, ‘Never (0 - 60 min a week)’, ‘Sometimes (60 - 150 minutes a week)’, ‘Often (more than 150 minutes per week’)
gender	Ordinal	‘Male’, ‘Female’, ‘Non-binary’
diet	Ordinal	‘I do not follow any diet’, ‘Vegetarian diet (you refrain from the consumption of meat (red meat, poultry, seafood, insects and the flesh of any other animal)’, ‘Pescatarian diet (your diet includes fish and seafood, but not the flesh of other animals)’, ‘Vegan diet (you refrain from eating any animal products)’, “I follow a diet that isn’t listed here”, ‘Prefer not to say’
residence.area	Ordinal	‘Urban’, ‘Suburban’, ‘Rural’, ‘Prefer not to say’
FNS	Double	Food neophobia scale level; calculated as average of Food Neophobia items
GMS	double	Scores on general attitude towards genetic modification; calculated as average of attitude towards GM foods answers
NKS		Nutrition knowledge Score, calculated based on how many statements were correctly identified as True or False.

Field name	Data type	Description
FRL1-20/FRLH1-8		Food related lifestyle related answers to Likert questions. Food-Related Lifestyle: Is split-up into 7 different dimensions • Importance of product info • Price criteria • Interest in cooking • Convenience • Cultural/social • Taste • Health

Data preparation

1. Initial setup + importing data

```
setwd("C:/Users/marct/Documents/Rstudio projects/Thesis")
```

```
library(dplyr)
library(ggplot2)
library(ggsci)
library(viridis)
library(psych)
library(tidyr)
library(lavaan)
```

```
# Create a base theme and use the nature color scheme
base_theme <- theme_minimal(base_size = 14) +
  theme(
    text         = element_text(family = ""),
    plot.title   = element_text(face = "bold", size = 16, hjust = 0.5),
    axis.title   = element_text(size = 14),
    axis.text    = element_text(color = "black"),
    panel.grid.major = element_line(color = "grey90"),
    panel.grid.minor = element_blank(),
    legend.position  = "bottom",
    legend.title     = element_text(face = "italic")
  )

npg_colors <- pal_npg("nrc", alpha = 1)(5) # Using Nature color scheme!
```

As explained in the methodology section of the report, 10 different versions were shown to participants, with different order and/or bioengineered logo present or absent. Versions 1 through 5 did not have the bioengineered logo present, while versions 6 through 10 did have the bioengineered logo present. Apart from the bioengineered logo, the packaging showed between versions 1 through 5, and 6 through 10 were the

same, but differently ordered. Before analysis, this has to be corrected for in the dataset. There were 4 Nutrition/Health focused statements regarding: (1) Protein, (2) Fiber, (3) Gut, and (4) Heart-health. The following orders of the scenarios were used:

- V1/6: 1, 2, 3, 4
- V2/7: 2, 4, 1, 3
- V3/8: 3, 1, 4, 2
- V4/9: 4, 3, 2, 1
- V5/10: 1, 4, 2, 3

Code is written to import and order the answers correctly, finally a single dataframe is formed

```
dta <- rio::import("Results.xlsx")
dta$duration <- dta$`Completion time` - dta$`Start time`
dta <- dta[,-c(2:11)] #removing time stamp, prolific ID, 'mock' screening questions, etc. Var
iables that are not used in analysis

dta$bioengineered <- ifelse(dta$Version >= 6, "Bioengineered logo", "No logo") # Creating var
iable weather the version contained the 'bioengineered logo' (1) or not (0)

# Import the different versions into dataframes, making sure each of these dataframes (dtaV1_
6, etc.) have the same scenario order within the dataframe
dtaV1_6 <- dta[dta$Version %in% c(1,6),]
dtaV2_7 <- dta[dta$Version %in% c(2,7),]
dtaV3_8 <- dta[dta$Version %in% c(3,8),]
dtaV4_9 <- dta[dta$Version %in% c(4,9),]
dtaV5_10 <- dta[dta$Version %in% c(5,10),]

# Changing the column names of each of these versions, to reflect the different order of the
scenarios. Afterwards, all dataframes should have the same order of scenarios
colnames(dtaV1_6) <- c('version', 'likeliness.plantbased', 'age','physical.act','main.grocer
y.shopper','gender','people.in.household', 'living.with.children','household.income','diet',
'living.region', 'education.level', 'familiarity.plantbased','NP1', 'NP2', 'NP3', 'NP4', 'NP
5', 'NP6','NP7','NP8','NP9','NP10','FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL
8','FRL9','FRL10','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FR
L20','FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8','GM1','GM2','GM3','GM
4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM13','NRK1','NRK2','NRK3','NRK4','NRK
5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK1
7','NRK18','NRK19','NRK20',
'att.chk1','art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
'att.chk2','art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
'att.chk3','art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
'att.chk4','art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
'reason.for.trying','nutyeast.association','yeastbm.association','mycelium.assocation','yeast
p.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HR
S8','gm.likely','gm.benefits','duration', 'bioengineered')

colnames(dtaV2_7) <- c('version', 'likeliness.plantbased', 'age','physical.act','main.grocer
y.shopper','gender','people.in.household', 'living.with.children','household.income','diet',
'living.region', 'education.level', 'familiarity.plantbased','NP1', 'NP2', 'NP3', 'NP4', 'NP
5', 'NP6','NP7','NP8','NP9','NP10','FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL
8','FRL9','FRL10','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FR
L20','FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8','GM1','GM2','GM3','GM
4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM13','NRK1','NRK2','NRK3','NRK4','NRK
5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK1
7','NRK18','NRK19','NRK20',
'att.chk2','art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
'att.chk4','art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
'att.chk1','art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
'att.chk3','art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
'reason.for.trying','nutyeast.association','yeastbm.association','mycelium.assocation','yeast
p.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HR
S8','gm.likely','gm.benefits','duration', 'bioengineered')

colnames(dtaV3_8) <- c('version', 'likeliness.plantbased', 'age','physical.act','main.grocer
y.shopper','gender','people.in.household', 'living.with.children','household.income','diet',
'living.region', 'education.level', 'familiarity.plantbased','NP1', 'NP2', 'NP3', 'NP4', 'NP
```

```
5', 'NP6','NP7','NP8','NP9','NP10','FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL
8','FRL9','FRL10','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FR
L20','FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8','GM1','GM2','GM3','GM
4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM13','NRK1','NRK2','NRK3','NRK4','NRK
5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK1
7','NRK18','NRK19','NRK20',
'att.chk3','art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
'att.chk1','art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
'att.chk4','art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
'att.chk2','art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
'reason.for.trying','nutyeast.association','yeastbm.association','mycelium.assocation','yeast
p.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HR
S8','gm.likely','gm.benefits','duration', 'bioengineered')

colnames(dtaV4_9) <- c('version', 'likeliness.plantbased', 'age','physical.act','main.grocer
y.shopper','gender','people.in.household', 'living.with.children','household.income','diet',
'living.region', 'education.level', 'familiarity.plantbased','NP1', 'NP2', 'NP3', 'NP4', 'NP
5', 'NP6','NP7','NP8','NP9','NP10','FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL
8','FRL9','FRL10','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FR
L20','FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8','GM1','GM2','GM3','GM
4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM13','NRK1','NRK2','NRK3','NRK4','NRK
5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK1
7','NRK18','NRK19','NRK20',
'att.chk4','art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
'att.chk3','art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
'att.chk2','art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
'att.chk1','art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
'reason.for.trying','nutyeast.association','yeastbm.association','mycelium.assocation','yeast
p.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HR
S8','gm.likely','gm.benefits','duration', 'bioengineered')

colnames(dtaV5_10) <- c('version', 'likeliness.plantbased', 'age','physical.act','main.grocer
y.shopper','gender','people.in.household', 'living.with.children','household.income','diet',
'living.region', 'education.level', 'familiarity.plantbased','NP1', 'NP2', 'NP3', 'NP4', 'NP
5', 'NP6','NP7','NP8','NP9','NP10','FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL
8','FRL9','FRL10','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FR
L20','FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8','GM1','GM2','GM3','GM
4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM13','NRK1','NRK2','NRK3','NRK4','NRK
5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK1
7','NRK18','NRK19','NRK20',
'att.chk1','art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
'att.chk4','art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
'att.chk2','art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
'att.chk3','art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
'reason.for.trying','nutyeast.association','yeastbm.association','mycelium.assocation','yeast
p.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HR
S8','gm.likely','gm.benefits','duration', 'bioengineered')

# Merge dataframes back into one dataset, now with the same order of scenarios (protein, fibe
r, gut, heart-health)
dta <- rbind(dtaV1_6, dtaV2_7, dtaV3_8, dtaV4_9, dtaV5_10)

# Removing other unneeded variables
dta <- dta[,-5] # remove 'main grocery shopper' variable
```

```
str(dta)
head(dta)
```

2. Data cleaning & recoding

2.1 Cleaning and subsetting data

```
# Turning most variables into factors, for further analysis
dta$likeliness.plantbased <- factor(dta$likeliness.plantbased,levels=c('Not at all likely',
'Not so likely', 'Somewhat likely', 'Very likely', 'Extremely likely'))

dta <- dta %>%
  mutate(across(.cols = !c(living.region,version, age, att.chk1,att.chk2,att.chk3,att.chk4,re
ason.for.trying,nutyeast.association,yeastbm.association,mycelium.assocation,yeastp.associati
on,mycoprotein.association),  # columns to convert and exclude in conversion to factor
  .fns = as.factor))
```

In the survey, a custom screener was used to only select for participant that are at least ‘somewhat likely’ to try plant-based meat in the future. In the full survey dataset, people that answered ‘not at all likely’ or ‘not so likely’ are screened out, but their data (until being screened out) is still in the dataset. The following plot shows how likely the total amount of participants were to try plant-based meats.

```
# Make bar plot of likeliness to try plant-based meat
p0 <- ggplot(dta, aes(x = likeliness.plantbased)) +
  geom_bar(fill=npg_colors[3]) +
  scale_fill_npg() +
  labs(title = '', x = '', y = '# of Participants') +
  base_theme

p0
```

```
#ggsave("PBM_likeliness.png", plot = p0, width = 6, height = 4, dpi = 300)

# Calculate and print: (1) How many particpants in total were recruited, (2) The percentages
of participants being not interested in plant-based meat, and interested in plant-based meat
counts <- table(dta$likeliness.plantbased)
cbind((prop.table(table(dta$likeliness.plantbased))))

not_interested <- 100*sum(counts[c("Not at all likely", "Not so likely")]) / sum(counts)
interested <- 100-not_interested

print(paste0('Total amount of participants: ', sum(counts)))
print(paste0('Not likely: ', round(not_interested,2),'%'))
print(paste0('At least somewhat likely: ', round(interested,2),'%'))
```

2.2 Adjusting variables

Now that variable names are sorted, some tuning/cleaning/creating of variables needs to happen:

- Screened out participants need to be removed from the dataset
- Age should be categorized in brackets based on generation

- Most variables should be converted to factors, for analysis. In this process, some category names are shortened (ie. replacing ‘Urban (e.g., city center or densely populated area)’ with just ‘Urban’. This is mostly for visualization in bar charts at a further stage.
- Likert question answers should be converted to numeric scores (1-5) instead of text ‘Not at all likely’, etc.
- Some variables should be calculated from the Likert questions (Food Neophobia scores, Food Related Lifestyle dimension scores, GM attitude scores)

```
# Removing screened out participants and categorising participants in age brackets
dta <- dta %>%
  filter(!is.na(gender)) %>%                 # Filter out all people that did not reply the 'What is your gender?' question with 'Male', 'Female', 'Non-binary' or 'Prefer not to say', since those are the people that were screened out
  mutate(generation = case_when(             # Divide people's age into categories, for presenting the results in the report the exact ages are not of importance
    age >= 13 & age <= 28 ~ "Gen Z",
    age >= 29 & age <= 44 ~ "Millennials",
    age >= 45 & age <= 60 ~ "Gen X",
    age >= 61 & age <= 79 ~ "Boomers",
    TRUE ~ NA_character_  # for ages outside these ranges (there were none outside of these ages)
  ))

### Factorize variables of interest. Explicitely state levels in factor command to make sure in ggplot the order is logical
dta$generation <- factor(dta$generation,levels=c('Gen Z', 'Millennials', 'Gen X', 'Boomers'))
dta$education.level <- factor(dta$education.level,levels=c("Don't know / not applicable", 'No formal qualifications', 'Prefer not to say', 'Secondary education (e.g. GED/GCSE)', 'High school diploma / A-levels', 'Technical/community college', 'Undergraduate degree (BA/BSc/other)','Graduate degree (MA/MSc/MPhil/other)', 'Doctorate degree (PhD/other)' ))
dta$education.level.general <- ifelse(dta$education.level == "High school diploma / A-levels","Low",ifelse(dta$education.level == "Technical/community college", "Medium", ifelse(dta$education.level == "Undergraduate degree (BA/BSc/other)", "Medium", ifelse(dta$education.level == "Graduate degree (MA/MSc/MPhil/other)", "High", ifelse(dta$education.level == "Doctorate degree (PhD/other)", "High", "Other")))))

dta$education.level.general <- factor(dta$education.level.general, levels = c("Low", "Medium", "High"))


dta$household.income <- gsub("\u00A0", " ", dta$household.income, fixed = TRUE) # Remove ASCII spaces present in some ranges
dta$household.income <- factor(dta$household.income,levels=c('Less than $60,000','$60,000 - $79,999','$80,000 - $99,999','$100,000 - $119,999','$120,000 - $139,999','$140,000 - $159,999','$160,000 - $179,999','More than $180,000','Prefer not to say'))

dta$diet <- ifelse(dta$diet == 'I do not follow any diet', 'No diet',
                   ifelse(dta$diet == 'Vegetarian diet (you refrain from the consumption of meat (red meat, poultry, seafood, insects and the flesh of any other animal)', 'Vegetarian',
                          ifelse(dta$diet == 'Pescatarian diet (your diet includes fish and seafood, but not the flesh of other animals)', 'Pescetarian',
                                 ifelse(dta$diet == 'Vegan diet (you refrain from eating any animal products)', 'Vegan',
                                        ifelse(dta$diet == "I follow a diet that isn't listed here", 'Other','Prefer not to say'))))
                   )

dta$living.region <- ifelse(dta$living.region == 'Urban (e.g., city center or densely populated area)', 'Urban',
                   ifelse(dta$living.region == 'Suburban (e.g., residential area outside a city)', 'Suburban',
                          ifelse(dta$living.region == 'Rural (e.g. countryside, farmland, or small village)', 'Rural', 'Prefer not to say')))
```

```
dta$living.region <- factor(dta$living.region,levels=c('Urban','Suburban','Rural'))

dta$physical.act <- factor(dta$physical.act,levels=c('Never (0 - 60 minutes per week)','Somet
imes (60 - 150 minutes per week)','Often (more than 150 minutes per week)'))
```

```
# Converting Likert question answers to numerical values
likert_map <- c(
  "Strongly disagree" = 1,
  "Disagree" = 2,
  "Neutral" = 3,
  "Agree" = 4,
  "Extremely agree" = 5,
  "Strongly agree" = 5,

  "Very artificial" = 1,
  "Artificial" = 2,
  "Neither artificial nor natural" = 3,
  "Natural" = 4,
  "Very natural" = 5,

  "Very unhealthy" = 1,
  "Unhealthy" = 2,
  "Neither healthy nor unhealthy" = 3,
  "Somewhat healthy" = 4,
  "Very healthy" = 5,

  "Extremely unlikely" = 1,
  "Very unlikely" = 1,
  "Somewhat unlikely" = 2,
  "Unlikely" = 2,
  "Neutral" = 3,
  "Neither likely nor unlikely" = 3,
  "Somewhat likely" = 4,
  "Likely" = 4,
  "Very likely" = 5,
  "Extremely likely" = 5,

  "Much less appealing" = 1,
  "Somewhat less appealing" = 2,
  "Not sure / no opinion" = 3,
  "Somewhat more appealing" = 4,
  "Much more appealing" = 5,

  "True" = 1,
  "False" = 0,
  "Not sure" = 2,

  "Much less appealing" = 1,
  "Somewhat less appealing" = 2,
  "Not sure / no opinion" = 3,
  "Somewhat more appealing" = 4,
  "Much more appealing" = 5
)

likert_questions <- c('NP1', 'NP2', 'NP3', 'NP4', 'NP5', 'NP6','NP7','NP8','NP9','NP10',
                      'FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL8','FRL9','FRL1
0','FRL11','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FRL20',
                      'FRLH1','FRLH2','FRLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8',
                      'GM1','GM2','GM3','GM4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM
12','GM13',
```

```
                      'NRK1','NRK2','NRK3','NRK4','NRK5','NRK6','NRK7','NRK8','NRK9','NRK1
0','NRK11','NRK12','NRK13','NRK14','NRK15','NRK16','NRK17','NRK18','NRK19','NRK20',
                      'art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
                      'art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
                      'art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
                      'art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
                      'HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HRS8',
                      'gm.likely')

dta_likert <- dta[likert_questions]

# Apply the likert map to the data
dta_likert_num <- data.frame(lapply(dta_likert, function(x) {
  if(is.factor(x) || is.character(x)) {
    # Map character/factor responses to numeric values
    as.numeric(likert_map[as.character(x)])
  } else {
    x
  }
}))
```

```
### Use MICE to add missing values
library(mice)
library(VIM)  # for missing data patterns

# Reverse code: FRL2 and FRL17
FRL_rev_items <- c("FRL9")#"FRL2","FRL17")
NP_rev_items <- c("NP1","NP4","NP6","NP9","NP10")
GM_rev_items <- c("GM4","GM7","GM9","GM10","GM11", "GM12", "GM13")

# Define min and max values of Likert scale
min_val <- 1
max_val <- 5

# Reverse values indicated before
dta_likert_num[FRL_rev_items] <- (max_val + min_val) - dta_likert_num[FRL_rev_items]
dta_likert_num[NP_rev_items] <- (max_val + min_val) - dta_likert_num[NP_rev_items]
dta_likert_num[GM_rev_items] <- (max_val + min_val) - dta_likert_num[GM_rev_items]

# Check missing data pattern
VIM::aggr(dta_likert_num, col = c('navyblue','red'), numbers = TRUE, sortVars = TRUE)
```

```
# Multiple imputation using MICE
# Use predictive mean matching (pmm) for Likert scales 5-7 points
imp <- mice(dta_likert_num, method = 'pmm', m = 5, seed = 123)

# Complete the datasets
dta_likert_num <- complete(imp, action = 1)#, include = TRUE)
```

Several of the measured variables still need to be obtained from the combined answers of Likert-scale questions. This will be done in the next section:

```
# Making some identifiers for useful subsets of the data
NP_vars <- c('NP1', 'NP2', 'NP3', 'NP4', 'NP5', 'NP6','NP7','NP8','NP9','NP10')
FRL_vars <- c('FRL1', 'FRL2','FRL3','FRL4','FRL5','FRL6','FRL7','FRL8','FRL9','FRL10','FRL1
1','FRL12','FRL13','FRL14','FRL15','FRL16','FRL17','FRL18','FRL19','FRL20','FRLH1','FRLH2','F
RLH3','FRLH4','FRLH5','FRLH6','FRLH7','FRLH8')
GM_vars <- c('GM1','GM2','GM3','GM4','GM5','GM6','GM7','GM8','GM9','GM10','GM11','GM12','GM1
3')
NRK_vars <- c('NRK1','NRK2','NRK3','NRK4','NRK5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK1
1','NRK12','NRK13','NRK14','NRK15','NRK16','NRK17','NRK18','NRK19','NRK20')

# Specifying the 7 different dimensions for clustering later
#FRL_product_vars <- c('FRL1','FRL8','FRL20')
#FRL_price_vars <- c('FRL7','FRL15','FRL13') #'FRL13')
#FRL_cooking_vars <- c('FRL2','FRL9') #,'FRL17')
#FRL_convenience_vars <- c('FRL11','FRL6') #,'FRL3')
#FRL_cultural_vars <- c('FRL14', 'FRL5', 'FRL16', 'FRL10')
#FRL_taste_vars <- c('FRL12', 'FRL19', 'FRL18')
#FRL_health_vars <- c('FRLH1', 'FRLH6', 'FRLH8','FRLH4')
#FRL_health_gfi_vars <- c('FRLH2', 'FRLH5', 'FRLH3','FRLH7')

### Food-Related Lifestyle
FRL_product_vars <- c('FRL1','FRL8','FRL20')
FRL_price_vars <- c('FRL7','FRL15','FRL13') #'FRL13')
FRL_cooking_vars <- c('FRL2','FRL9') #,'FRL17')
FRL_convenience_vars <- c('FRL11','FRL6') #,'FRL3')
FRL_cultural_vars <- c('FRL14', 'FRL5', 'FRL16', 'FRL10')
FRL_taste_vars <- c('FRL12', 'FRL19', 'FRL18')
FRL_health_vars <- c('FRLH1', 'FRLH6', 'FRLH8','FRLH4')
FRL_health_gfi_vars <- c('FRLH5', 'FRLH7')

# Calculating scores and cronsbach alpha of each dimension
dta_likert_num$FRL_product <- rowSums(dta_likert_num[FRL_product_vars]) / length(FRL_product_
vars)
FRL_prod_alpha <- alpha(dta_likert_num[FRL_product_vars])$total$raw_alpha

dta_likert_num$FRL_price <- rowSums(dta_likert_num[FRL_price_vars]) / length(FRL_price_vars)
FRL_price_alpha <- alpha(dta_likert_num[FRL_price_vars])$total$raw_alpha

dta_likert_num$FRL_cooking <- rowSums(dta_likert_num[FRL_cooking_vars]) / length(FRL_cooking_
vars)
FRL_cooking_alpha <- alpha(dta_likert_num[FRL_cooking_vars])$total$raw_alpha

dta_likert_num$FRL_convenience <- rowSums(dta_likert_num[FRL_convenience_vars]) / length(FRL_
convenience_vars)
FRL_convenience_alpha <- alpha(dta_likert_num[FRL_convenience_vars])$total$raw_alpha

dta_likert_num$FRL_cultural <- rowSums(dta_likert_num[FRL_cultural_vars]) / length(FRL_cultur
al_vars)
FRL_cultural_alpha <- alpha(dta_likert_num[FRL_cultural_vars])$total$raw_alpha

dta_likert_num$FRL_taste <- rowSums(dta_likert_num[FRL_taste_vars]) / length(FRL_taste_vars)
FRL_taste_alpha <- alpha(dta_likert_num[FRL_taste_vars])$total$raw_alpha

dta_likert_num$FRL_health <- rowSums(dta_likert_num[FRL_health_vars]) / length(FRL_health_var
s)
```

```
FRL_health_alpha <- alpha(dta_likert_num[FRL_health_vars])$total$raw_alpha

dta_likert_num$FRL_health_gfi <- rowSums(dta_likert_num[FRL_health_gfi_vars]) / length(FRL_he
alth_gfi_vars)
FRL_health_gfi_alpha <- alpha(dta_likert_num[FRL_health_gfi_vars])$total$raw_alpha

FRL_dimensions <- c("FRL_product", "FRL_price", "FRL_cooking", "FRL_convenience", "FRL_cultur
al", "FRL_taste", "FRL_health", "FRL_health_gfi")

### Food Neophobia (High score is highly food neophobic)

# Calculating scores
dta_likert_num$FNS <- rowSums(dta_likert_num[NP_vars]) / length(NP_vars)

### Genetic Modification attitude vars (high score is positive attitude towards GM)
# Calculating scores
dta_likert_num$GMS <- rowSums(dta_likert_num[GM_vars]) / length(GM_vars)

### Nutrition Knowledge
NRK_vars <- c('NRK1','NRK2','NRK3','NRK4','NRK5','NRK6','NRK7','NRK8','NRK9','NRK10','NRK1
1','NRK12','NRK13','NRK14','NRK15','NRK16','NRK17','NRK18','NRK19','NRK20')
correct_answers <- c(0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)

# Calculate knowledge score for each participant
dta_likert_num$NKS <- 0

for(i in 1:20) {
  item_col <- NRK_vars[i]
  dta_likert_num$NKS <- dta_likert_num$NKS +
    ifelse(dta_likert_num[[item_col]] == correct_answers[i], 1, 0)
}

dta_likert_num$NKS <- 100*dta_likert_num$NKS / length(NRK_vars) # Turn NKS into percentage co
rrect answers

### Trust
trustA_vars <- c('trustA1', 'trustA2', 'trustA3', 'trustA4')
trustB_vars <- c('trustB1', 'trustB2', 'trustB3', 'trustB4')
trustC_vars <- c('trustC1', 'trustC2', 'trustC3', 'trustC4')

trust1_vars <- c('trustA1','trustB1','trustC1')
trust2_vars <- c('trustA2','trustB2','trustC2')
trust3_vars <- c('trustA3','trustB3','trustC3')
trust4_vars <- c('trustA4','trustB4','trustC4')

# Calculating scores
dta_likert_num$trust1 <- rowSums(dta_likert_num[trust1_vars]) / 3
dta_likert_num$trust2 <- rowSums(dta_likert_num[trust2_vars]) / 3
dta_likert_num$trust3 <- rowSums(dta_likert_num[trust3_vars]) / 3
dta_likert_num$trust4 <- rowSums(dta_likert_num[trust4_vars]) / 3
```

```
demographics <- c("age", "generation", "gender", "household.income", "education.level","educa
tion.level.general", "living.region", "diet", "physical.act", "people.in.household","living.w
ith.children")
reg_vars <- c('art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
                      'art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
                      'art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
                      'art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1',
                      'gm.likely','FNS', 'GMS','trust1','trust2','trust3','trust4')

# Create one dataframe again, including all useful variables and including whether the versio
n contained 'bioengineered'
dta_tot <- cbind(dta[demographics],dta_likert_num[c(reg_vars,FRL_dimensions)],dta["bioenginee
red"])

# Create dataset with only 'bioengineered' group data, and other data
dta_BE <- subset(dta_tot, bioengineered == "Bioengineered logo")
dta_nBE <- subset(dta_tot, bioengineered == "No logo")
```

3. Descriptive statistics

3.1 Sample characteristics

Start by plotting some demographics

```
library(patchwork)
library(gridExtra)

p1 <- ggplot(dta_BE, aes(x = generation)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Generation (BE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p2 <- ggplot(dta_nBE, aes(x = generation)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Generation (nBE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p3 <- ggplot(dta_BE, aes(x = gender)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Gender (nBE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p4 <- ggplot(dta_nBE, aes(gender)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Gender (nBE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p5 <- ggplot(dta_BE, aes(x = household.income)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Household income (BE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p6 <- ggplot(dta_nBE, aes(x=household.income)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Household income (nBE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p7 <- ggplot(dta_BE, aes(x = gender)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Gender (BE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))

p8 <- ggplot(dta_nBE, aes(gender)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Gender (nBE)', y = 'Number of people') +
  base_theme + theme(axis.text.x = element_text(angle = 45, hjust = 1))


grid.arrange(p1, p2, p3, p4, ncol = 4)
```

Calculate percentages of demographic aspects, to put in a table in my report.

```
cbind(prop.table(table(dta_tot$generation))*100)
cbind(prop.table(table(dta_tot$gender))*100)
cbind(prop.table(table(dta_tot$household.income))*100)
cbind(prop.table(table(dta_tot$education.level))*100)
cbind(prop.table(table(dta_tot$living.region))*100)
cbind(prop.table(table(dta_tot$diet))*100)
cbind(prop.table(table(dta_tot$physical.act))*100)
cbind(prop.table(table(dta$people.in.household))*100)
cbind(prop.table(table(dta$living.with.children))*100)


cbind(prop.table(table(dta_tot$education.level.general))*100)
```

```
cbind(prop.table(table(dta_nBE$generation))*100)
cbind(prop.table(table(dta_BE$generation))*100)

cbind(prop.table(table(dta_nBE$gender))*100)
cbind(prop.table(table(dta_BE$gender))*100)

cbind(prop.table(table(dta_nBE$household.income))*100)
cbind(prop.table(table(dta_BE$household.income))*100)

cbind(prop.table(table(dta_nBE$education.level.general))*100)
cbind(prop.table(table(dta_BE$education.level.general))*100)

cbind(prop.table(table(dta_nBE$living.region))*100)
cbind(prop.table(table(dta_BE$living.region))*100)

cbind(prop.table(table(dta_nBE$diet))*100)
cbind(prop.table(table(dta_BE$diet))*100)

#cbind(prop.table(table(dta_nBE$physical.act))*100)
#cbind(prop.table(table(dta_BE$physical.act))*100)

cbind(prop.table(table(dta_nBE$people.in.household))*100)
cbind(prop.table(table(dta_BE$people.in.household))*100)

cbind(prop.table(table(dta_nBE$living.with.children))*100)
cbind(prop.table(table(dta_BE$living.with.children))*100)
```

Descriptive statistics (mean, sd, cronsbach alpha when relevant)

```
# Exploring numeric variables: Food Neophobia, GM attitude, FRL dimensions, Nutrition Knowled
ge
print("FNS")
summary(dta_likert_num$FNS)
sd(dta_likert_num$FNS)
alpha(dta_likert_num[NP_vars])$total$raw_alpha

print("GMS")
summary(dta_likert_num$GMS)
sd(dta_likert_num$GMS, na.rm=TRUE)
alpha(dta_likert_num[GM_vars])$total$raw_alpha

print("NK")
summary(dta_likert_num$NKS)
sd(dta_likert_num$NKS, na.rm=TRUE)

# Product
print("FRL_product")
summary(dta_likert_num$FRL_product)
sd(dta_likert_num$FRL_product, na.rm=TRUE)
print(FRL_prod_alpha)

# Price
print("FRL_price")
summary(dta_likert_num$FRL_price)
sd(dta_likert_num$FRL_price, na.rm=TRUE)
print(FRL_price_alpha)

# Cooking
print("FRL_cooking")
summary(dta_likert_num$FRL_cooking)
sd(dta_likert_num$FRL_cooking, na.rm=TRUE)
print(FRL_cooking_alpha)

# Convenience
print("FRL_convenience")
summary(dta_likert_num$FRL_convenience)
sd(dta_likert_num$FRL_convenience, na.rm=TRUE)
print(FRL_convenience_alpha)

# Cultural
print("FRL_cultural")
summary(dta_likert_num$FRL_cultural)
sd(dta_likert_num$FRL_cultural, na.rm=TRUE)
print(FRL_cultural_alpha)

# Taste
print("FRL_taste")
summary(dta_likert_num$FRL_taste)
sd(dta_likert_num$FRL_taste, na.rm=TRUE)
print(FRL_taste_alpha)

# health
print("FRL_taste")
summary(dta_likert_num$FRL_health)
```

```
sd(dta_likert_num$FRL_health, na.rm=TRUE)
print(FRL_health_alpha)

# health gfi
print("FRL_taste")
summary(dta_likert_num$FRL_health_gfi)
sd(dta_likert_num$FRL_health_gfi, na.rm=TRUE)
print(FRL_health_gfi_alpha)
```

For situational variables:

```
# For all trust etc vars build a loop
# Creating some subsets that will be useful
scenario_vars <- c('art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4',
                       'art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3',
                       'art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2',
                       'art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1')

sc1_vars <- c('art1','health1','trustA1','trustB1','trustC1','wtt1','wtb1')
sc2_vars <- c('art2','health2','trustA2','trustB2','trustC2','wtt2','wtb2')
sc3_vars <- c('art3','health3','trustA3','trustB3','trustC3','wtt3','wtb3')
sc4_vars <- c('art4','health4','trustA4','trustB4','trustC4','wtt4','wtb4')

art_vars <- c('art1', 'art2', 'art3', 'art4')
health_vars <- c('health1','health2', 'health3', 'health4')
trustA_vars <- c('trustA1', 'trustA2', 'trustA3', 'trustA4')
trustB_vars <- c('trustB1', 'trustB2', 'trustB3', 'trustB4')
trustC_vars <- c('trustC1', 'trustC2', 'trustC3', 'trustC4')
trustABC_vars <- c('trust1', 'trust2', 'trust3', 'trust4')
wtt_vars <- c('wtt1', 'wtt2', 'wtt3', 'wtt4')
wtb_vars <- c('wtb1', 'wtb2', 'wtb3', 'wtb4')

# Build a loop for computing cronsbach alpha when applicable
for (var in c(art_vars, health_vars, trustABC_vars, wtt_vars, wtb_vars)) {
  # Get summary stats
  s <- summary(dta_nBE[[var]])

  print(var)
  print("No bioengineered label")

  # Print median (3rd element of summary vector)
  print(paste0("Mean: ", s["Mean"]))

  # Print standard deviation
  print(paste0("SD: ", sd(dta_nBE[[var]], na.rm = TRUE)))

  # Print range
  print(paste0("Range: ", s["Min."], " - ", s["Max."]))

  # If it's a trust variable, compute alpha for the corresponding trio
  if (grepl("^trust[1-4]$", var)) {
    # Extract the digit
    idx <- sub("trust", "", var)
    t_vars <- paste0(c("trustA", "trustB", "trustC"), idx)

    # Compute Cronbach's alpha
    alf_out <- alpha(dta_nBE[t_vars])
    alpha_val <- alf_out$total$raw_alpha

    print(paste0("Cronbach's alpha for ", paste(t_vars, collapse = ", "), ": ", round(alpha_v
al, 2)))
  }
}

for (var in c(art_vars, health_vars, trustABC_vars, wtt_vars, wtb_vars)) {
  # Get summary stats
```

```
  s <- summary(dta_BE[[var]])

  print(var)
  print("Bioengineered label present")

  # Print median (3rd element of summary vector)
  print(paste0("Mean: ", s["Mean"]))

  # Print standard deviation
  print(paste0("SD: ", sd(dta_BE[[var]], na.rm = TRUE)))

  # Print range
  print(paste0("Range: ", s["Min."], " - ", s["Max."]))

  # If it's a trust variable, compute alpha for the corresponding trio
  if (grepl("^trust[1-4]$", var)) {
    # Extract the digit
    idx <- sub("trust", "", var)
    t_vars <- paste0(c("trustA", "trustB", "trustC"), idx)

    # Compute Cronbach's alpha
    alf_out <- alpha(dta_BE[t_vars])
    alpha_val <- alf_out$total$raw_alpha

    print(paste0("Cronbach's alpha for ", paste(t_vars, collapse = ", "), ": ", round(alpha_v
al, 2)))
  }
}
```

```
# Exploring ordinal/categorical variables by plotting them
plots <- vector("list", length(demographics))
names(plots) <- demographics

for (v in demographics) {
  print(cbind(table(dta[v])))

  dat <- dta

  # Special filter for diet
  if (v == "diet") {
    dat <- dat %>% filter(diet != "No diet")
  }

  p <- ggplot(dat, aes_string(x = v)) +
              geom_bar(fill='skyblue') +
              labs(
                title = "Survey responses",
                x     = gsub("\\.", " ", tools::toTitleCase(v)),
                y     = "Number of people"
              ) +
              theme_minimal() +
              theme(axis.text.x = element_text(angle = 45, hjust = 1)) + scale_fill_npg()

  plots[[v]] <- p
}

plots
```

```
#dta_NP_numeric$NP_category <- ifelse(dta_NP_numeric$FN < 25, "neophilic", "neophobic")
#dta_likert_num$FNS <- scale(dta_likert_num$FNS)
#dta_likert_num$GMS <- scale(dta_likert_num$GMS)

ggplot(dta_likert_num, aes(x = FNS)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Level of Food Neophobia', y = 'Number of people') +
  base_theme
```

```
ggsave("fns.png", width = 6, height = 4, dpi = 300)

ggplot(dta_likert_num %>% filter (!is.na(GMS)), aes(x = GMS)) +
  geom_bar(fill=npg_colors[2]) +
  scale_fill_npg() +
  labs(title = '', x = 'General attitude towards GM', y = 'Number of people') +
  base_theme
```

```
ggsave("gms.png", width = 6, height = 4, dpi = 300)

ggplot(dta_likert_num, aes(x = NKS)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Level of Nutrition knowledge', y = 'Number of people') +
  base_theme
```

```
ggsave("nks.png", width = 6, height = 4, dpi = 300)


ggplot(dta_likert_num, aes(x = FRL_product)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'FRL: Importance of product', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_price)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'FRL: Importance of price', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_cooking)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of cooking', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_convenience)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of convenience', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_cultural)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of cultural', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_taste)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of taste', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_health)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of health', y = 'Number of people') +
  base_theme
```

```
ggplot(dta_likert_num, aes(x = FRL_health_gfi)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Importance of gfi health', y = 'Number of people') +
  base_theme
```

```
### Creating correlation table of relevant variables, to get an idea of the correlation of va
riables

# Merging perceived naturalness, healthiness, trust, and wtt/wtb to reduce amount of variable
s, while still getting an idea
# Creating separate dataset for the correlation table
cor_vars <- c("FNS","GMS","NKS", "gm.likely") # Variables relevant for use in the correlation
table
dta_cor <- dta_likert_num[cor_vars]

dta_cor$trust_avg <- rowSums(dta_tot[trustABC_vars]) / 4
dta_cor$art_avg <- rowSums(dta_tot[art_vars]) / 4
dta_cor$health_avg <- rowSums(dta_tot[health_vars]) / 4
dta_cor$pi <- rowSums(dta_tot[c(wtt_vars,wtb_vars)]) / 8 # Merge Willingness To Try and Willi
ngness To Buy in one metric for correlation matrix
dta_cor2 <- cbind(dta_cor,dta_likert_num[FRL_dimensions])

apaTables::apa.cor.table(dta_cor2)
```

```
### Making another correlation table to see if demographics correlate with variables, to deci
de whether to include them in the linear mixed effects model
dta_cor[c('NKS','gm.likely','pi')] <- NULL
dta_cor$wtt <- rowSums(dta_tot[wtt_vars]) / 4
dta_cor$wtb <- rowSums(dta_tot[wtb_vars]) / 4
dta_cor$age <- as.numeric(dta_tot$age)

levels(dta_tot$gender)
table(dta_tot$gender, as.numeric(dta_tot$gender))

levels(dta_tot$living.region)
table(dta_tot$living.region, as.numeric(dta_tot$living.region))

dta_cor$gender <- as.numeric(dta_tot$gender)
dta_cor$household.income <- as.numeric(dta_tot$household.income)
dta_cor$education.level <- as.numeric(dta_tot$education.level)
dta_cor$living.region <- as.numeric(dta_tot$living.region)

apaTables::apa.cor.table(dta_cor)
```

```
ggplot(dta_cor2, aes(x = trust_avg)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Average trust', y = 'Number of people') +
  base_theme
```

```
ggsave("trust.png", width = 6, height = 4, dpi = 300)

ggplot(dta_cor2, aes(x = art_avg)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Average perceived naturalness', y = 'Number of people') +
  base_theme
```

```
ggsave("nat.png", width = 6, height = 4, dpi = 300)

ggplot(dta_cor2, aes(x = health_avg)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Average perceived healthiness', y = 'Number of people') +
  base_theme
```

```
ggsave("health.png", width = 6, height = 4, dpi = 300)

ggplot(dta_cor, aes(x = wtt)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Average willingness to try', y = 'Number of people') +
  base_theme
```

```
ggsave("wtt.png", width = 6, height = 4, dpi = 300)

ggplot(dta_cor, aes(x = wtb)) +
  geom_bar(fill=npg_colors[1]) +
  scale_fill_npg() +
  labs(title = '', x = 'Average willingness to buy', y = 'Number of people') +
  base_theme
```

```
ggsave("wtb.png", width = 6, height = 4, dpi = 300)
```

```
# Create plot to show averages of situational variables
averages <- c(mean(dta_cor2$trust_avg),mean(dta_cor2$health_avg),mean(dta_cor2$art_avg),mean
(dta_cor$wtt),mean(dta_cor$wtb))
avg_names <- c('Trust', 'Perceived healthiness', 'Perceived naturalness', 'Willingness to tr
y', 'Willingness to buy')

avgs <- t(as.data.frame(averages))
colnames(avgs) <- avg_names

avgs_long <- pivot_longer(as.data.frame(avgs),
                          cols = everything(),
                          names_to = "Measure",
                          values_to = "Average")

# Plot
ggplot(avgs_long, aes(x = Measure, y = Average, fill = Measure)) +
  geom_bar(stat = "identity", alpha=0.8) +
  scale_fill_npg() +
  labs(title = '', x = 'Measure', y = 'Average') +
  base_theme + theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "none") +  # Remove legend since fill is redundant
  ylim(0, 5)
```

```
ggsave("averages.png", width = 10, height = 7, dpi = 300)
```

4. Hypothesis testing

4.1 Importing libraries and preparing data

```
library(lme4)
library(mediation)
library(lmerTest)
library(emmeans)
library(performance)
library(see)
library(car)

### Need this somewhere later
detach_package <- function(pkg, character.only = FALSE)
{
  if(!character.only)
  {
    pkg <- deparse(substitute(pkg))
  }
  search_item <- paste("package", pkg, sep = ":")
  while(search_item %in% search())
  {
    detach(search_item, unload = TRUE, character.only = TRUE)
  }
}

# Add some participant identifier for use in models later
t <- as.data.frame(1:length(dta_tot$gender))

dta_tot$id <- t$`1:length(dta_tot$gender)`

# add age, gender, education level

# Converting the 'wide' dataframe to a 'long' one, as needed by statistical models analysing
mixed effects (measuring situational varibles in multiple scenarios)

dta_long <- dta_tot %>%
  #filter(!(gender == 'Non-binary')) %>%
  pivot_longer(
    cols = matches("^(art|trustA|trustB|trustC|health|wtt|wtb)\\d+$"),
    names_to  = c("measure", "scenario"),
    names_pattern = "^([A-Za-z]+)(\\d+)$",
    values_to = "value"
  ) %>%
  pivot_wider(
    names_from  = measure,
    values_from = value
  )

dta_long$scenario <- as.numeric(dta_long$scenario)
dta_long$claim_type <- as.factor(ifelse(dta_long$scenario >= 3,"Health claim", "Ingredient cl
aim"))
dta_long$NHC <- as.factor(ifelse(dta_long$scenario == 1,"NHC1", ifelse(dta_long$scenario ==
2,"NHC2",ifelse(dta_long$scenario == 3,"NHC3","NHC4"))))
```

```
dta_long$bioengineered <- relevel(factor(dta_long$bioengineered), ref = "No logo")
dta_long$trust_tot <- (dta_long$trustA + dta_long$trustB + dta_long$trustC)/3
table(is.na(dta_long$health))
```

Testing the influence of demographics as covariates:

```
model1 <- lmer(health ~ (1|id), data=dta_long)
summary(model1)
model1a <- lmer(wtt ~ as.numeric(age) + gender + education.level + living.region + (1|id), da
ta=dta_long)

anova(model1,model1a)
```

Direct effects on willingness to try:

```
# Model 1: Control variables only
model1 <- lmer(wtt ~ (1|id), data=dta_long)
summary(model1)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtt ~ bioengineered + (1|id), data=dta_long)

model3 <- lmer(wtt ~ claim_type + (1|id), data=dta_long)

model4 <- lmer(wtt ~ scale(health) + (1|id), data=dta_long)

model5 <- lmer(wtt ~ scale(art) + (1|id), data=dta_long)

model6 <- lmer(wtt ~ scale(trust_tot) + (1|id), data=dta_long)

model7 <- lmer(wtt ~ scale(FNS) + (1|id), data=dta_long)

summary(model2)
summary(model3)
summary(model4)
summary(model5)
summary(model6)
summary(model7)

# Test effects
#anova(model1, model1a)  # Doublecheck that demographics are not needed as control

anova(model1, model2)  # H1: Bioengineered logo reduces willingness to try
anova(model1, model3)  # Different health claims affect willingness to try differently
anova(model1, model4)  # Logo effect varies by claim type (interaction)
anova(model1, model5)  # Logo effect varies by claim type (interaction)
anova(model1, model6)  # Logo effect varies by claim type (interaction)
anova(model1, model7)  # Logo effect varies by claim type (interaction)
```

Direct effects on willingness to buy:

```
# Model 1: Control variables only
model1 <- lmer(wtb ~ (1|id), data=dta_long)
summary(model1)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtb ~ bioengineered + (1|id), data=dta_long)

model3 <- lmer(wtb ~ claim_type + (1|id), data=dta_long)

model4 <- lmer(wtb ~ scale(health) + (1|id), data=dta_long)

model5 <- lmer(wtb ~ scale(art) + (1|id), data=dta_long)

model6 <- lmer(wtb ~ scale(trust_tot) + (1|id), data=dta_long)

model7 <- lmer(wtb ~ scale(FNS) + (1|id), data=dta_long)

summary(model2)
summary(model3)
summary(model4)
summary(model5)
summary(model6)
summary(model7)

# Test effects
#anova(model1, model1a)  # Doublecheck that demographics are not needed as control

anova(model1, model2)  # H1: Bioengineered logo reduces willingness to try
anova(model1, model3)  # Different health claims affect willingness to try differently
anova(model1, model4)  # Logo effect varies by claim type (interaction)
anova(model1, model5)  # Logo effect varies by claim type (interaction)
anova(model1, model6)  # Logo effect varies by claim type (interaction)
anova(model1, model7)  # Logo effect varies by claim type (interaction)
```

Effect of claim_type on perceived healthiness

```
# Model 1: Control variables only
model1 <- lmer(health ~ (1|id), data=dta_long)
summary(model1)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(health ~ claim_type + (1|id), data=dta_long)
#model2 <- lmer(health ~ bioengineered + FNS + GMS + (1|id), data=dta_long)
summary(model2)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
```

Effect of bioengineered logo on perceived naturalness

```
# Model 1: Control variables only
model1 <- lmer(art ~ (1|id), data=dta_long)

summary(model1)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(art ~ bioengineered + (1|id), data=dta_long)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
```

Influence of bioengineered logo on trust

```
# Model 1: Control variables only
model1 <- lmer(trust_tot ~ (1|id), data=dta_long)
summary(model1)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(trust_tot ~ bioengineered + (1|id), data=dta_long)
summary(model2)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
```

4.2.2 Testing moderation effects

(1) Logo on (trust –> wtt/wtb)

```
#
model0 <- lmer(trust_tot ~ bioengineered +  (1|id), data=dta_long)
summary(model0)

# Model 1: Control variables only
model1 <- lmer(wtt ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtt ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtt ~ bioengineered + scale(trust_tot) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtt ~ bioengineered * scale(trust_tot) + (1|id), data=dta_long) #bioengineered
*claim_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)

# Fit an ordinary least-squares model with the same fixed effects to check colinearity
lm_fixed <- lm(wtt ~ bioengineered * scale(trust_tot), data = dta_long)

# Compute VIFs
vif(lm_fixed)

library(effects)

### Code based on Regression Analysis in R.Rmd (Tudelft, 2025), aided by ChatGPT

# 1. Compute just the interaction effect
eff <- effect(
  term  = "bioengineered:scale(trust_tot)",
  mod   = model4,
  xlevels = list(       # specify two levels of each predictor
    bioengineered  = levels(dta_long$bioengineered),
    scale.trust_tot = c(-2, 2)  # for example, ±2 SD
  )
)

# 2. Coerce to data.frame
eff_df <- as.data.frame(eff)

# eff_df now has exactly four rows:
#   bioengineered | scale.trust_tot | fit  | lower | upper

# 3. Plot directly (no further subsetting needed)
library(ggplot2)
```

```
ggplot(eff_df, aes(
    x     = trust_tot,
    y     = fit,
    linetype = bioengineered
  )) +
  geom_line(color = npg_colors[3], size = 1) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
  scale_linetype_manual(
    values = c("solid", "dotted"),
    labels = c("No logo", "Logo")
  ) +
  labs(
    x     = "Standardized Trust",
    y     = "Predicted Willingness to Try",
    linetype = NULL
  ) +
  base_theme
```

```
ggsave("trust_mod.png", width = 9, height = 6, dpi = 300)
```

```
# Model 1: Control variables only
model1 <- lmer(wtb ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtb ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtb ~ bioengineered + scale(trust_tot) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtb ~ bioengineered * scale(trust_tot) + (1|id), data=dta_long) #bioengineered
*claim_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)

# Fit an ordinary least-squares model with the same fixed effects to check colinearity
lm_fixed <- lm(wtb ~ bioengineered * scale(trust_tot), data = dta_long)

# Compute VIFs
vif(lm_fixed)
```

(2) Logo on (perceived healthiness –> wtt/wtb)

```
#
model0 <- lmer(trust_tot ~ bioengineered +  (1|id), data=dta_long)
summary(model0)

# Model 1: Control variables only
model1 <- lmer(wtt ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtt ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtt ~ bioengineered + scale(health) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtt ~ bioengineered * scale(health) + (1|id), data=dta_long) #bioengineered*cl
aim_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)

# 1. Fit an ordinary least-squares model with the same fixed effects
lm_fixed <- lm(wtt ~ bioengineered * scale(health), data = dta_long)

# 2. Compute VIFs
vif(lm_fixed)

# 1. Compute just the interaction effect
eff <- effect(
  term  = "bioengineered:scale(health)",
  mod   = model4,
  xlevels = list(       # specify two levels of each predictor
    bioengineered  = levels(dta_long$bioengineered),
    scale.health = c(-2, 2)  # for example, ±2 SD
  )
)

# 2. Coerce to data.frame
eff_df <- as.data.frame(eff)

# eff_df now has exactly four rows:
#   bioengineered | scale.trust_tot | fit  | lower | upper

# 3. Plot directly (no further subsetting needed)
library(ggplot2)

ggplot(eff_df, aes(
    x     = health,
    y     = fit,
    linetype = bioengineered
```

```
  )) +
  geom_line(color = npg_colors[3], size = 1) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
  scale_linetype_manual(
    values = c("solid", "dotted"),
    labels = c("No logo", "Logo")
  ) +
  labs(
    x     = "Standardized perceived healthiness",
    y     = "Predicted Willingness to Try",
    linetype = NULL
  ) +
  base_theme
```

```
ggsave("health_mod.png", width = 9, height = 6, dpi = 300)
```

```
# Model 1: Control variables only
model1 <- lmer(wtb ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtb ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtb ~ bioengineered + scale(health) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtb ~ bioengineered * scale(health) + (1|id), data=dta_long) #bioengineered*cl
aim_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)

# 1. Fit an ordinary least-squares model with the same fixed effects
lm_fixed <- lm(wtb ~ bioengineered * scale(health), data = dta_long)

# 2. Compute VIFs
vif(lm_fixed)
```

3. **Logo on (perceived healthiness –> wtt/wtb)**

```
# Model 1: Control variables only
model1 <- lmer(wtb ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtb ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtb ~ bioengineered + scale(art) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtb ~ bioengineered * scale(art) + (1|id), data=dta_long) #bioengineered*claim
_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)

# 1. Fit an ordinary least-squares model with the same fixed effects
lm_fixed <- lm(wtb ~ bioengineered * scale(art), data = dta_long)

# 2. Compute VIFs
vif(lm_fixed)
```

```
# calculate bonferroni ps to put in report, and to lower the risk of type I errors
p_vals <- c(
  p_trust = 0.000145,      # logo × trust
  p_health = 0.00345,      # logo × healthiness
  p_naturalness = 0.0758   # logo × naturalness
)

# Apply the Benjamini–Hochberg FDR correction
p_adj <- p.adjust(p_vals, method = "bonferroni")

# Combine into a table for easy viewing
results_df <- data.frame(
  interaction   = names(p_vals),
  p_uncorrected = p_vals,
  p_fdr         = p_adj
)

print(results_df)
```

Food neophobia moderation effects

```
# Model 1: Control variables only
model1 <- lmer(wtt ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtt ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtt ~ bioengineered + scale(FNS) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtt ~ bioengineered * scale(FNS) + (1|id), data=dta_long) #bioengineered*claim
_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)
vif(model4)
```

```
# Model 1: Control variables only
model1 <- lmer(wtb ~ (1|id), data=dta_long)

# Model 2: Adding direct effect of bioengineered logo
model2 <- lmer(wtb ~ bioengineered  + (1|id), data=dta_long)
summary(model2)

# Model 3: Add claim main effect
model3 <- lmer(wtb ~ bioengineered + scale(FNS) +  (1|id), data=dta_long)
summary(model3)

# Model 4: Add interaction
model4 <- lmer(wtb ~ bioengineered * scale(FNS) + (1|id), data=dta_long) #bioengineered*claim
_type = bioengineered*claim_type + bioengineered + claim_type in lmer!
summary(model4)

# Test effects
anova(model1, model2)  # H1: Bioengineered logo reduces perceived healthiness
anova(model2, model3)  # Different health claims affect perceived healthiness to try differen
tly***
anova(model3, model4)  # Logo effect varies by claim type (interaction)
vif(model4)
```

4.2.2 Mediation effects

Perceived healthiness as mediator for claim_type –> willingness to try

```r
#dta_long$health[is.na(dta_long$health)] <- median(dta_long$health, na.rm = TRUE)
#dta_long$health[is.na(dta_long$health)] <- mean(dta_long$health, na.rm = TRUE)
library(mediation)
library(lme4)
detach_package(lmerTest) # Needed for some reason to let mediate accept lmer models

# Scale perceived healthiness for comparability
dta_long$health_z <- scale(dta_long$health)

# 1. Mediator model: trust regressed on bioengineered label with random intercepts for subjec
t
mod.M <- lme4::lmer(health_z ~ claim_type + (1|id), data = dta_long)

# 2. Outcome model: willingness_to_try regressed on label_type and trust
#    with the same random structure
mod.Y <- lme4::lmer(wtt ~ claim_type + health_z + (1|id), data = dta_long)

library(lmerTest)
summary(lmer(wtt ~ claim_type + health_z + (1|id), data = dta_long))
summary(mod.M)
summary(mod.Y)

# 3. Mediation analysis
set.seed(30102025)
med.out <- mediation::mediate(mod.M, mod.Y,
                   treat    = "claim_type",
                   mediator = "health_z",
                   sims     = 1000)

# 4. Summary of direct, indirect, and total effects
summary(med.out)
```

5 Exploratory analyses

```
library(dplyr)
library(stringr)
library(ggplot2)

# Perform some data separation to be able to analyse the variables that were collected with t
ick boxes

exp_vars <- c('reason.for.trying','nutyeast.association','yeastbm.association','mycelium.asso
cation','yeastp.association','mycoprotein.association','HRS1','HRS2','HRS3','HRS4','HRS5','HR
S6','HRS7','HRS8','gm.likely','gm.benefits','duration', 'bioengineered')

pos_terms <- c('Healthy','Natural', 'Sustainable', 'Innovative', 'Ethical')
neg_terms <- c('Fungal','Processed', 'Lab-grown', 'Artificial','Unhealthy')
neutral_terms <- c('Not from plant or animal', 'None of the above', 'Animal-based', 'Plant-ba
sed (made from plants)')

dta_comparisons <- cbind(dta['reason.for.trying'], dta['gm.benefits'], dta_tot['FNS'],dta_tot
['GMS'], dta_tot['gm.likely'])

# Split semicolon-separated responses and count frequencies
analyze_multiple_choice <- function(data_column) {
  # Split each response and count individual reasons
  split_reasons <- data_column %>%
    str_split(";") %>%
    unlist() %>%
    str_trim() %>% # Remove whitespace
    .[. != ""] %>% # Remove empty strings
    .[. != "" & !is.na(.)]  # Remove NA

  # Count frequencies
  reason_counts <- table(split_reasons)

  # Convert to dataframe
  results <- data.frame(
    Reason = names(reason_counts),
    Count = as.numeric(reason_counts)
  ) %>%
    arrange(desc(Count))

  # Calculate percentages (excluding "not interested")
  #not_interested_count <- sum(str_detect(data_column, "No, I'm not interested"))
  #interested_responses <- length(data_column) - not_interested_count

  results$Percentage <- 100*results$Count / (length(data_column))

  return(results)
}

# Create professional visualization
create_reason_chart <- function(results_df,color) {
  plot_data <- results_df %>%
    #filter(!str_detect(Reason, "not interested")) %>%
    mutate(Reason = str_wrap(Reason, 30))
```

```
  ggplot(plot_data, aes(x = reorder(Reason, Percentage), y = Percentage)) +
    geom_col(fill = npg_colors[color], alpha = 0.8) +
    coord_flip() +
    labs(
      title = "",
      x = "Reason",
      y = "Percentage of Respondents (%)"
    ) +
    base_theme
}
```

```r
library(dplyr)
library(tidyr)
library(ggplot2)
library(stringr)
library(forcats)

# Order the reasons for trying the product based on food neophobia score
# Total participants
N_total <- length(dta_comparisons$FNS)

# Explode reasons and assign tertiles
dta_long <- dta_comparisons %>%
  mutate(id = row_number(),
         FNS_tertile = ntile(FNS, 3),
         FNS_tertile = factor(FNS_tertile,
                              levels = 1:3,
                              labels = c("Low","Medium","High"))) %>%
  dplyr::select(id, reason.for.trying, FNS_tertile) %>%
  separate_rows(reason.for.trying, sep = ";") %>%
  mutate(reason = str_trim(reason.for.trying)) %>%
  filter(reason != "")

# 2. Compute total percent per reason
reason_totals <- dta_long %>%
  count(reason, name = "n_total") %>%
  mutate(pct_total = n_total / n_distinct(dta_long$id) * 100) %>%
  arrange(pct_total)

# 3. Define ordered factor levels
ordered_reasons <- reason_totals$reason

# 4. Compute cell percentages
reason_tertile_pct <- dta_long %>%
  count(reason, FNS_tertile) %>%
  mutate(pct_all = n / n_distinct(dta_long$id) * 100,
         reason = factor(reason, levels = ordered_reasons))

# 5. Plot sorted stacked bars
ggplot(reason_tertile_pct, aes(x = reason, y = pct_all, fill = FNS_tertile)) +
  geom_col(color = "gray40") +
  coord_flip() +
  scale_fill_npg(name = "FNS Tertile") +
  labs(x = "Reason for Trying", y = "Percentage of All Participants (%)") +
  base_theme +
  theme(legend.position = "right")
```

```r
library(dplyr)
library(tidyr)
library(ggplot2)
library(stringr)

plot_reason_by_tertile <- function(data, score_col, reason_col, n_tiles = 3, wrap_width = 30)
{
  # data: data frame
  # score_col: string name of numeric score column (e.g., "FNS")
  # reason_col: string name of semicolon-delimited reason column (e.g., "reason.for.trying")
  # n_tiles: number of tiles (default 3 for tertiles)
  # wrap_width: character wrap width for reason labels

  # 1. Total participants
  N_total <- length(data[score_col])

  # 2. Explode reasons and assign tiles
  long_df <- data %>%
    mutate(id = row_number(),
           score = ntile(.data[[score_col]], 3),
           score = factor(score, levels = 1:3, labels = c("Low","Medium","High"))) %>%
    dplyr::select(id, all_of(reason_col), score) %>%
    separate_rows(.data[[reason_col]], sep = ";") %>%
    mutate(reason = str_trim(.data[[reason_col]])) %>%
    filter(reason != "")

  # 3. Compute total percent per reason and order ascending
  reason_totals <- long_df %>%
    count(reason, name = "n_total") %>%
    mutate(pct_total = n_total /  n_distinct(dta_long$id) * 100) %>%
    arrange(pct_total)

  # 4. Define ordered factor levels
  ordered_reasons <- reason_totals$reason

  # 5. Compute cell percentages
  plot_df <- long_df %>%
    count(reason, score) %>%
    mutate(pct_all = n /  n_distinct(dta_long$id) * 100,
           reason = factor(reason, levels = ordered_reasons))

  # 6. Generate plot
  ggplot(plot_df, aes(x = reason, y = pct_all, fill = score)) +
    geom_col(color = "gray40") +
    coord_flip() +
    scale_fill_npg(name = paste0(score_col, " Score")) +
    labs(
      x = "Reason",
      y = "Percentage of All Participants (%)",
      title = paste("")
    ) +
    base_theme +
    theme(legend.position = "right") + ylim(0,100)
}
```

```
dta_comparisons$age <- dta_tot$age

# For Food Neophobia Score tertiles:
plot_reason_by_tertile(dta_comparisons, score_col = "gm.likely", reason_col = "reason.for.try
ing")
```

```
ggsave("gm_likely_reasons.png", width = 9, height = 5, dpi = 300)

plot_reason_by_tertile(dta_comparisons, score_col = "FNS", reason_col = "reason.for.trying")
```

```
ggsave("FNS_likely_reasons.png", width = 9, height = 5, dpi = 300)

plot_reason_by_tertile(dta_comparisons, score_col = "gm.likely", reason_col = "gm.benefits")
```

```
ggsave("gmbenefits_reasons.png", width = 14, height = 5, dpi = 300)

plot_reason_by_tertile(dta_comparisons, score_col = "age", reason_col = "reason.for.trying")
```

```
ggsave("gmbenefits_reasons.png", width = 14, height = 5, dpi = 300)
```

Before running the other exploratory analyses, it was gauged whether people would want to try the product, and if so what the reason for trying would be.

```
dta_comparisons <- cbind(dta$reason.for.trying, dta_tot$FNS,dta_tot$GMS, dta_tot$gm.likely)

reason.for.trying_counts <- analyze_multiple_choice(dta$reason.for.trying)
reason.for.trying_counts$Percentage <- 100*reason.for.trying_counts$Count / (length(dta$reaso
n.for.trying))
reason.for.trying_counts$Percentage

reason.for.trying_plot <- create_reason_chart(reason.for.trying_counts,3)
reason.for.trying_plot
```

```
ggsave("Reason_to_try.png", plot = reason.for.trying_plot, width = 8, height = 4, dpi = 300)
```

5.1 Word association analyses

```
nutyeast_counts <- analyze_multiple_choice(dta$nutyeast.association)
create_reason_chart(nutyeast_counts,3)
```

```
yeastbm_counts <- analyze_multiple_choice(dta$yeastbm.association)
create_reason_chart(yeastbm_counts,3)
```

```
mycelium_counts <- analyze_multiple_choice(dta$mycelium.assocation)
create_reason_chart(mycelium_counts,3)
```

```
yeastp_counts <- analyze_multiple_choice(dta$yeastp.association)
create_reason_chart(yeastp_counts,3)
```

```
mycoprotein_counts <- analyze_multiple_choice(dta$mycoprotein.association)
create_reason_chart(mycoprotein_counts,3)
```

```
nutyeast_counts <- nutyeast_counts %>% arrange(Reason)
yeastbm_counts <- yeastbm_counts %>% arrange(Reason)
mycelium_counts <- mycelium_counts %>% arrange(Reason)
yeastp_counts <- yeastp_counts %>% arrange(Reason)
mycoprotein_counts <- mycoprotein_counts %>% arrange(Reason)

nutyeast_counts
yeastbm_counts
mycelium_counts
yeastp_counts
mycoprotein_counts
```

```
df_wide <- tibble(
   Reason     = nutyeast_counts$Reason,
   Mycoprotein            = mycoprotein_counts$Percentage,
   Nutritional_Yeast      = nutyeast_counts$Percentage,
   Yeast_Biomass          = yeastbm_counts$Percentage,
   Mycelium          = mycelium_counts$Percentage,
   Yeast_Protein          = yeastp_counts$Percentage
 )

df_long <- df_wide %>%
  pivot_longer(
    cols      = -Reason,        # all columns except Reason
    names_to  = "Name",         # new column for product names
    values_to = "Percentage"    # values go here
  ) %>%
  mutate(
    Name = str_replace_all(Name, "_", " ")  # clean up underscores
  )

library(forcats)
library(scales)
base <- 28
association_plot <- ggplot(df_long, aes(
    x    = Percentage,
    y    = fct_rev(fct_reorder(Reason, Percentage)),  # reverse order so first reason on top
    fill = Name
  )) +
  geom_col(position = position_dodge(width = 0.8),  # group bars side by side
           width = 0.7) +
  scale_fill_npg() +
  labs(
    title = "",
    x     = "Percentage of Respondents",
    y     = "",
    fill  = "Product Name"
  ) +
  coord_flip() +
  base_theme +
  theme(

    plot.title    = element_text(size = base*1.4, face = "bold"),
    axis.title.y = element_text(size=base),
    axis.text.y   = element_text(size = base),
    axis.text.x = element_text(size = base, angle = 45, hjust = 1),
    legend.title  = element_text(size = base*1.1),
    legend.text   = element_text(size = 0.9*base)
  )

ggsave("Association plot.png", plot = association_plot, width = 30, height = 15, dpi = 300)
```

```
nps_dta <- as.data.frame(t(df_wide[-1]))  # exclude first column from transpose
colnames(nps_dta) <- df_wide[[1]]

nps_dta$pos_avg <- rowSums(nps_dta[pos_terms])/length(pos_terms)
nps_dta$neg_avg <- rowSums(nps_dta[neg_terms])/length(neg_terms)
nps_dta$nps <- nps_dta$pos_avg - nps_dta$neg_avg
nps_dta$Name <- rownames(nps_dta)

nps_plot <- ggplot(nps_dta, aes(
    x    = fct_reorder(Name, nps),  # order bars by NPS if you like
    y    = nps,
    fill = Name
  )) +
  geom_col(color = "white", size = 0.2, width = 0.7,alpha=0.8) +
  scale_fill_npg() +               # discrete NPG palette
  labs(
    title = "",
    x     = "Product Name",
    y     = "Net Positivity Score",
    fill  = "Product Name"
  ) +
  coord_flip() +                   # horizontal bars
  base_theme +
  theme(
    axis.text.y   = element_text(size = 12),     # product labels
    axis.text.x   = element_text(size = 12),     # NPS labels
    axis.title.x  = element_text(size = 10),
    axis.title.y  = element_text(size = 10),
    legend.position = "none"                    # no legend needed
  )

ggsave("nps.png", plot = nps_plot, width = 6, height = 4, dpi = 300)
```

```
nps_dta
```

5.2 Risk/benefit analyses

```
HRS_vars <- c('HRS1','HRS2','HRS3','HRS4','HRS5','HRS6','HRS7','HRS8')
test <- cbind(dta_likert_num[HRS_vars],dta_tot[demographics],dta_likert_num['gm.likely'])
test$age <- as.numeric(test$age)
```

```
test <- test %>%
  mutate(
    age_tertile = ntile(age, 3),                        # create tertile groups
    age_tertile = factor(age_tertile,                   # make factor with labels
                         levels = 1:3,
                         labels = c("Low", "Medium", "High"))
  )

# Check the result
table(test$age_tertile, useNA="ifany")
test$age_tertile

test %>%
  group_by(age_tertile) %>%
  summarize(
    n = n(),
    age_min = min(age, na.rm = TRUE),
    age_max = max(age, na.rm = TRUE),
    age_median = median(age, na.rm = TRUE)
  )
```

```
aov_model <- aov(gm.likely ~ living.region , data = test)
summary(aov_model)
```

```
gm.benefits_counts <- analyze_multiple_choice(dta$gm.benefits)
create_reason_chart(gm.benefits_counts,3)
```

6 Clustering

Clustering of FRL, partly according to Bronso et al., building independent models for main things.

```
library(lavaan)
FRL_CFA <- scale(dta_likert_num[FRL_vars])

model_waysofshopping <- '
  product =~ FRL1 + FRL8 + FRL20
  price =~ FRL4
'


model_quality <-
'
  price_q =~ FRL7 + FRL13 + FRL15
  taste =~ FRL12 + FRL19 + FRL18
  health_prod_attr =~ FRLH1 + FRLH6 + FRLH8 + FRLH4
'


model_cookingmethods <-
'
  cooking =~ FRL2 + FRL9
  convenience =~ FRL11 + FRL6
'


model_purchasing_motives <-
'
  cultural =~ FRL14 + FRL5 + FRL16
  act =~ FRLH5 + FRLH7 + FRLH3
'


cfa_1 <- cfa(model_waysofshopping,
              data=FRL_CFA,
              estimator = "MLR",      # robust maximum likelihood
              missing = "FIML",       # full information ML
              std.lv = TRUE)          # standardize latent variances

cfa_2 <- cfa(model_quality,
              data=FRL_CFA,
              estimator = "MLR",      # robust maximum likelihood
              missing = "FIML",       # full information ML
              std.lv = TRUE)          # standardize latent variances

cfa_3 <- cfa(model_cookingmethods,
              data=FRL_CFA,
              estimator = "MLR",      # robust maximum likelihood
              missing = "FIML",       # full information ML
              std.lv = TRUE)          # standardize latent variances

cfa_4 <- cfa(model_purchasing_motives,
              data=FRL_CFA,
              estimator = "MLR",      # robust maximum likelihood
              missing = "FIML",       # full information ML
              std.lv = TRUE)          # standardize latent variances

#cfa_5 <- cfa(model_activity,
#              data=FRL_CFA,
#              estimator = "MLR",      # robust maximum likelihood
#              missing = "FIML",       # full information ML
```

```
#               std.lv = TRUE)          # standardize latent variances

#summary(cfa_1, fit.measures=TRUE)
#summary(cfa_2, fit.measures=TRUE)
#summary(cfa_3, fit.measures=TRUE)
#summary(cfa_4, fit.measures=TRUE)

shopping <- lavPredict(cfa_1)              # factor scores
quality <- lavPredict(cfa_2)
cooking <- lavPredict(cfa_3)
social <- lavPredict(cfa_4)
#activity <- lavPredict(cfa_5)

fitMeasures(cfa_1, c("chisq","df","rmsea","aic","tli","cfi"))
fitMeasures(cfa_2, c("chisq","df","rmsea","aic","tli","cfi"))
fitMeasures(cfa_3, c("chisq","df","rmsea","aic","tli","cfi"))
fitMeasures(cfa_4, c("chisq","df","rmsea","aic","tli","cfi"))

try <- as.data.frame(scale(as.data.frame(cbind(shopping,quality,cooking,social))))
```

CFA

Next, clustering can begin. To determine the ideal amount of clusters, three different metrics are often used. The silhouette score should be close to 1 (above 0.5 is good, above 0.25 is indication). wss is elbow plot. Gap stat..

```
c_dim <- scale(dta_tot[c(FRL_dimensions,"GMS","FNS")])
```

```
library(factoextra)
try <- cbind(try, as.data.frame(scale(dta_likert_num['FNS'])))
fviz_nbclust(try, kmeans, method = "silhouette",k.max=8) + base_theme + labs(title = "")
```

```
fviz_nbclust(try, kmeans, method = "silhouette",k.max=8) + base_theme + labs(title = "")
```

```
ggsave("silhouette.png", width = 10, height = 7, dpi = 300)

fviz_nbclust(try, kmeans, method = "wss",k.max=8) + base_theme + labs(title = "")
```

```
ggsave("elbow.png", width = 10, height = 7, dpi = 300)

fviz_nbclust(try, kmeans, method = "gap_stat",k.max=8) + base_theme + labs(title = "")
```

```
ggsave("gap.png", width = 10, height = 7, dpi = 300)
```

```
k <- 2 # set the number of clusters here
clusters <- kmeans(try, k)
FRL_cls <- cbind(try, cluster = clusters$cluster)

cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta$generation,
  education = dta$education.level
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)
chisq.test(cluster_profiles$cluster, cluster_profiles$generation)

factoextra::fviz_cluster(clusters, FRL_cls[, -which(colnames(FRL_cls) %in% "cluster") ],
                         geom = "point",
                         ellipse.type = "convex",
                         ggtheme = theme_bw())
```

```
FRL_cls_z <- as.data.frame(lapply(FRL_cls, scale))
dis = dist(FRL_cls_z)
sil = cluster::silhouette(clusters$cluster, dis)
plot(sil, border = NA)
```

```
cbind(FRL_cls[ which(sil[, 3] < 0), ], sil[,2:3][which( sil[,3] < 0), ])

# Cluster distributions
plot_data_long <- reshape2::melt(FRL_cls, id.vars = "cluster") # converting the data frame to
a 'long' format (see earlier explanation of the reshape2 package)
normalized_densities <- data.frame() # create empty data frame to store results

# Calculate normalized densities for each combination of variable and cluster
for (variable_name in unique(plot_data_long$variable)) {
    for (cluster_name in unique(plot_data_long$cluster)) {
        subset_data <- subset(plot_data_long, variable == variable_name & cluster == cluster_
name)
        dens <- density(subset_data$value, na.rm = TRUE) # Ensure to handle NAs if present
        normalized_density <- dens$y / sum(dens$y)
        temp_df <- data.frame(value = dens$x, density = normalized_density, variable = variab
le_name, cluster = cluster_name)
        normalized_densities <- rbind(normalized_densities, temp_df)
    }
}

# Create the ggplot object with relative density plots for each variable, faceted by variable
plt <- ggplot(normalized_densities, aes(x = value, y = density, colour = as.factor(cluster)))
+
    geom_line() +
    facet_wrap(~ variable, scales = "free") + # Adjust 'nrow' and 'ncol' based on the number
of variables
    theme_minimal() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          strip.background = element_rect(fill = "lightgrey"),
          legend.position = "bottom") +
    labs(colour = "Cluster")

# Print the plot
print(plt)
```

```
# Clean up the working space a bit
remove(plot_data_long, normalized_densities, variable_name, cluster_name, subset_data, dens,
normalized_density, temp_df, plt)

# Calculate mean scores per cluster
aggregate(. ~ cluster,
          data = FRL_cls,
          FUN = mean)

siz <- data.frame(cls = sort(unique(as.character(paste("Cluster", clusters$cluster)))),
                  frq = as.vector(table(clusters$cluster)))
siz$prp <- siz$frq / sum(siz$frq) * 100

ggplot(siz, aes(x = "", y = frq, fill = cls)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    theme_void() +
    scale_fill_discrete(name = "Clusters",
                        breaks = siz$cls,
                        labels = paste(siz$cls, "-", paste(round(siz$prp, 1), "%", sep =
""")))
```

```
model <- C50::C5.0(as.factor(cluster) ~.,
                   data = FRL_cls)
summary(model)

dta_reg <- dta %>%
  filter(!education.level %in% c("Don't know / not applicable", "Prefer not to say"))

dta_reg$generation.num <- as.numeric(factor(dta_reg$generation,
                                        levels = c("Gen Z","Millennials","Gen X","Boomers")))

dta_reg$education.level.num <- as.numeric(factor(dta_reg$education.level,
                                        levels = c("No formal qualifications", "Secondary edu
cation (e.g. GED/GCSE)", "High school diploma / A-levels", "Technical/community college", "Un
dergraduate degree (BA/BSc/other)","Graduate degree (MA/MSc/MPhil/other)", "Doctorate degree
(PhD/other)")))

apaTables::apa.cor.table(dta_reg[c('generation.num', 'education.level.num')])
```

```
k <- 3 # set the number of clusters here
clusters <- kmeans(try, k)
FRL_cls <- cbind(try, cluster = clusters$cluster)

cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta$generation,
  education = dta$education.level
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)

factoextra::fviz_cluster(clusters, FRL_cls[, -which(colnames(FRL_cls) %in% "cluster") ],
                         geom = "point",
                         ellipse.type = "convex",
                         ggtheme = theme_bw())
```

```
FRL_cls_z <- as.data.frame(lapply(FRL_cls, scale))
dis = dist(FRL_cls_z)
sil = cluster::silhouette(clusters$cluster, dis)
plot(sil, border = NA)
```

```
cbind(FRL_cls[ which(sil[, 3] < 0), ], sil[,2:3][which( sil[,3] < 0), ])

# Cluster distributions
plot_data_long <- reshape2::melt(FRL_cls, id.vars = "cluster") # converting the data frame to
a 'long' format (see earlier explanation of the reshape2 package)
normalized_densities <- data.frame() # create empty data frame to store results

# Calculate normalized densities for each combination of variable and cluster
for (variable_name in unique(plot_data_long$variable)) {
    for (cluster_name in unique(plot_data_long$cluster)) {
        subset_data <- subset(plot_data_long, variable == variable_name & cluster == cluster_
name)
        dens <- density(subset_data$value, na.rm = TRUE) # Ensure to handle NAs if present
        normalized_density <- dens$y / sum(dens$y)
        temp_df <- data.frame(value = dens$x, density = normalized_density, variable = variab
le_name, cluster = cluster_name)
        normalized_densities <- rbind(normalized_densities, temp_df)
    }
}

# Create the ggplot object with relative density plots for each variable, faceted by variable
plt <- ggplot(normalized_densities, aes(x = value, y = density, colour = as.factor(cluster)))
+
    geom_line() +
    facet_wrap(~ variable, scales = "free") + # Adjust 'nrow' and 'ncol' based on the number
of variables
    theme_minimal() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          strip.background = element_rect(fill = "lightgrey"),
          legend.position = "bottom") +
    labs(colour = "Cluster")

# Print the plot
print(plt)
```

```
# Clean up the working space a bit
remove(plot_data_long, normalized_densities, variable_name, cluster_name, subset_data, dens,
normalized_density, temp_df, plt)

# Calculate mean scores per cluster
aggregate(. ~ cluster,
          data = FRL_cls,
          FUN = mean)

siz <- data.frame(cls = sort(unique(as.character(paste("Cluster", clusters$cluster)))),
                  frq = as.vector(table(clusters$cluster)))
siz$prp <- siz$frq / sum(siz$frq) * 100

ggplot(siz, aes(x = "", y = frq, fill = cls)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    theme_void() +
    scale_fill_discrete(name = "Clusters",
                        breaks = siz$cls,
                        labels = paste(siz$cls, "-", paste(round(siz$prp, 1), "%", sep =
""")))
```

```
model <- C50::C5.0(as.factor(cluster) ~.,
                   data = FRL_cls)
summary(model)

dta_reg <- dta %>%
  filter(!education.level %in% c("Don't know / not applicable", "Prefer not to say"))

dta_reg$generation.num <- as.numeric(factor(dta_reg$generation,
                                        levels = c("Gen Z","Millennials","Gen X","Boomers")))

dta_reg$education.level.num <- as.numeric(factor(dta_reg$education.level,
                                        levels = c("No formal qualifications", "Secondary edu
cation (e.g. GED/GCSE)", "High school diploma / A-levels", "Technical/community college", "Un
dergraduate degree (BA/BSc/other)","Graduate degree (MA/MSc/MPhil/other)", "Doctorate degree
(PhD/other)")))

apaTables::apa.cor.table(dta_reg[c('generation.num', 'education.level.num')])
```

```
k <- 4 # set the number of clusters here
clusters <- kmeans(try, k)
FRL_cls <- cbind(try, cluster = clusters$cluster)

cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta$generation,
  education = dta$education.level
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)

factoextra::fviz_cluster(clusters, FRL_cls[, -which(colnames(FRL_cls) %in% "cluster") ],
                         geom = "point",
                         ellipse.type = "convex",
                         ggtheme = theme_bw())
```

```
FRL_cls_z <- as.data.frame(lapply(FRL_cls, scale))
dis = dist(FRL_cls_z)
sil = cluster::silhouette(clusters$cluster, dis)
plot(sil, border = NA)
```

```
cbind(FRL_cls[ which(sil[, 3] < 0), ], sil[,2:3][which( sil[,3] < 0), ])

# Cluster distributions
plot_data_long <- reshape2::melt(FRL_cls, id.vars = "cluster") # converting the data frame to
a 'long' format (see earlier explanation of the reshape2 package)
normalized_densities <- data.frame() # create empty data frame to store results

# Calculate normalized densities for each combination of variable and cluster
for (variable_name in unique(plot_data_long$variable)) {
    for (cluster_name in unique(plot_data_long$cluster)) {
        subset_data <- subset(plot_data_long, variable == variable_name & cluster == cluster_
name)
        dens <- density(subset_data$value, na.rm = TRUE) # Ensure to handle NAs if present
        normalized_density <- dens$y / sum(dens$y)
        temp_df <- data.frame(value = dens$x, density = normalized_density, variable = variab
le_name, cluster = cluster_name)
        normalized_densities <- rbind(normalized_densities, temp_df)
    }
}

# Create the ggplot object with relative density plots for each variable, faceted by variable
plt <- ggplot(normalized_densities, aes(x = value, y = density, colour = as.factor(cluster)))
+
    geom_line() +
    facet_wrap(~ variable, scales = "free") + # Adjust 'nrow' and 'ncol' based on the number
of variables
    theme_minimal() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          strip.background = element_rect(fill = "lightgrey"),
          legend.position = "bottom") +
    labs(colour = "Cluster") + base_theme

# Print the plot
print(plt)
```

```
# Clean up the working space a bit
remove(plot_data_long, normalized_densities, variable_name, cluster_name, subset_data, dens,
normalized_density, temp_df, plt)

# Calculate mean scores per cluster
aggregate(. ~ cluster,
          data = FRL_cls,
          FUN = mean)

siz <- data.frame(cls = sort(unique(as.character(paste("Cluster", clusters$cluster)))),
                  frq = as.vector(table(clusters$cluster)))
siz$prp <- siz$frq / sum(siz$frq) * 100

ggplot(siz, aes(x = "", y = frq, fill = cls)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    theme_void() +
    scale_fill_discrete(name = "Clusters",
                        breaks = siz$cls,
                        labels = paste(siz$cls, "-", paste(round(siz$prp, 1), "%", sep =
"""))) + base_theme
```

```
model <- C50::C5.0(as.factor(cluster) ~.,
                   data = FRL_cls)
summary(model)

dta_reg <- dta %>%
  filter(!education.level %in% c("Don't know / not applicable", "Prefer not to say"))

dta_reg$generation.num <- as.numeric(factor(dta_reg$generation,
                                        levels = c("Gen Z","Millennials","Gen X","Boomers")))

dta_reg$education.level.num <- as.numeric(factor(dta_reg$education.level,
                                        levels = c("No formal qualifications", "Secondary edu
cation (e.g. GED/GCSE)", "High school diploma / A-levels", "Technical/community college", "Un
dergraduate degree (BA/BSc/other)","Graduate degree (MA/MSc/MPhil/other)", "Doctorate degree
(PhD/other)")))

apaTables::apa.cor.table(dta_reg[c('generation.num', 'education.level.num')])
```

```
dta_tot <- cbind(dta_tot,clusters$cluster)
dta_tot
```

```
cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta_tot$generation,
  education = dta_tot$education.level.general,
  diet = dta_tot$diet,
  living.region = dta_tot$living.region,
  gender = dta_tot$gender,
  houshold_income = dta_tot$household.income,
  children = dta_tot$living.with.children,
  age = dta_tot$age,
  act = dta_tot$physical.act,
  diet = dta_tot$diet,
  gm = dta_tot$gm.likely
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$education)
chisq.test(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)
chisq.test(cluster_profiles$cluster, cluster_profiles$diet)
chisq.test(cluster_profiles$cluster, cluster_profiles$living.region)
chisq.test(cluster_profiles$cluster, cluster_profiles$gender)
chisq.test(cluster_profiles$cluster, cluster_profiles$household_income)
chisq.test(cluster_profiles$cluster, cluster_profiles$children)
chisq.test(cluster_profiles$cluster, cluster_profiles$age)
chisq.test(cluster_profiles$cluster, cluster_profiles$physical.act)
chisq.test(cluster_profiles$cluster, cluster_profiles$diet)

chisq.test(cluster_profiles$cluster, cluster_profiles$gm)

fisher.test(cluster_profiles$cluster, cluster_profiles$education,
            simulate.p.value = TRUE, B = 1e5)
# B = number of simulations; increase for more precision
```

```
k <- 5 # set the number of clusters here
clusters <- kmeans(try, k)
FRL_cls <- cbind(try, cluster = clusters$cluster)

cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta$generation,
  education = dta$education.level
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)

factoextra::fviz_cluster(clusters, FRL_cls[, -which(colnames(FRL_cls) %in% "cluster") ],
                         geom = "point",
                         ellipse.type = "convex",
                         ggtheme = theme_bw())
```

```
FRL_cls_z <- as.data.frame(lapply(FRL_cls, scale))
dis = dist(FRL_cls_z)
sil = cluster::silhouette(clusters$cluster, dis)
plot(sil, border = NA)
```

```
cbind(FRL_cls[ which(sil[, 3] < 0), ], sil[,2:3][which( sil[,3] < 0), ])

# Cluster distributions
plot_data_long <- reshape2::melt(FRL_cls, id.vars = "cluster") # converting the data frame to
a 'long' format (see earlier explanation of the reshape2 package)
normalized_densities <- data.frame() # create empty data frame to store results

# Calculate normalized densities for each combination of variable and cluster
for (variable_name in unique(plot_data_long$variable)) {
    for (cluster_name in unique(plot_data_long$cluster)) {
        subset_data <- subset(plot_data_long, variable == variable_name & cluster == cluster_
name)
        dens <- density(subset_data$value, na.rm = TRUE) # Ensure to handle NAs if present
        normalized_density <- dens$y / sum(dens$y)
        temp_df <- data.frame(value = dens$x, density = normalized_density, variable = variab
le_name, cluster = cluster_name)
        normalized_densities <- rbind(normalized_densities, temp_df)
    }
}

# Create the ggplot object with relative density plots for each variable, faceted by variable
plt <- ggplot(normalized_densities, aes(x = value, y = density, colour = as.factor(cluster)))
+
    geom_line() +
    facet_wrap(~ variable, scales = "free") + # Adjust 'nrow' and 'ncol' based on the number
of variables
    theme_minimal() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          strip.background = element_rect(fill = "lightgrey"),
          legend.position = "bottom") +
    labs(colour = "Cluster")

# Print the plot
print(plt)
```

```
# Clean up the working space a bit
remove(plot_data_long, normalized_densities, variable_name, cluster_name, subset_data, dens,
normalized_density, temp_df, plt)

# Calculate mean scores per cluster
aggregate(. ~ cluster,
          data = FRL_cls,
          FUN = mean)

siz <- data.frame(cls = sort(unique(as.character(paste("Cluster", clusters$cluster)))),
                  frq = as.vector(table(clusters$cluster)))
siz$prp <- siz$frq / sum(siz$frq) * 100

ggplot(siz, aes(x = "", y = frq, fill = cls)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    theme_void() +
    scale_fill_discrete(name = "Clusters",
                        breaks = siz$cls,
                        labels = paste(siz$cls, "-", paste(round(siz$prp, 1), "%", sep =
""")))
```

```
model <- C50::C5.0(as.factor(cluster) ~.,
                   data = FRL_cls)
summary(model)

dta_reg <- dta %>%
  filter(!education.level %in% c("Don't know / not applicable", "Prefer not to say"))

dta_reg$generation.num <- as.numeric(factor(dta_reg$generation,
                                        levels = c("Gen Z","Millennials","Gen X","Boomers")))

dta_reg$education.level.num <- as.numeric(factor(dta_reg$education.level,
                                        levels = c("No formal qualifications", "Secondary edu
cation (e.g. GED/GCSE)", "High school diploma / A-levels", "Technical/community college", "Un
dergraduate degree (BA/BSc/other)","Graduate degree (MA/MSc/MPhil/other)", "Doctorate degree
(PhD/other)")))

apaTables::apa.cor.table(dta_reg[c('generation.num', 'education.level.num')])
```

```
k <- 6 # set the number of clusters here
clusters <- kmeans(try, k)
FRL_cls <- cbind(try, cluster = clusters$cluster)

cluster_profiles <- data.frame(
  cluster = clusters$cluster,
  generation = dta$generation,
  education = dta$education.level
)

# Analyze demographic differences between clusters
table(cluster_profiles$cluster, cluster_profiles$generation)
chisq.test(cluster_profiles$cluster, cluster_profiles$education)

factoextra::fviz_cluster(clusters, FRL_cls[, -which(colnames(FRL_cls) %in% "cluster") ],
                         geom = "point",
                         ellipse.type = "convex",
                         ggtheme = theme_bw())
```

```
FRL_cls_z <- as.data.frame(lapply(FRL_cls, scale))
dis = dist(FRL_cls_z)
sil = cluster::silhouette(clusters$cluster, dis)
plot(sil, border = NA)
```

```
cbind(FRL_cls[ which(sil[, 3] < 0), ], sil[,2:3][which( sil[,3] < 0), ])

# Cluster distributions
plot_data_long <- reshape2::melt(FRL_cls, id.vars = "cluster") # converting the data frame to
a 'long' format (see earlier explanation of the reshape2 package)
normalized_densities <- data.frame() # create empty data frame to store results

# Calculate normalized densities for each combination of variable and cluster
for (variable_name in unique(plot_data_long$variable)) {
    for (cluster_name in unique(plot_data_long$cluster)) {
        subset_data <- subset(plot_data_long, variable == variable_name & cluster == cluster_
name)
        dens <- density(subset_data$value, na.rm = TRUE) # Ensure to handle NAs if present
        normalized_density <- dens$y / sum(dens$y)
        temp_df <- data.frame(value = dens$x, density = normalized_density, variable = variab
le_name, cluster = cluster_name)
        normalized_densities <- rbind(normalized_densities, temp_df)
    }
}

# Create the ggplot object with relative density plots for each variable, faceted by variable
plt <- ggplot(normalized_densities, aes(x = value, y = density, colour = as.factor(cluster)))
+
    geom_line() +
    facet_wrap(~ variable, scales = "free") + # Adjust 'nrow' and 'ncol' based on the number
of variables
    theme_minimal() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          strip.background = element_rect(fill = "lightgrey"),
          legend.position = "bottom") +
    labs(colour = "Cluster")

# Print the plot
print(plt)
```

```
# Clean up the working space a bit
remove(plot_data_long, normalized_densities, variable_name, cluster_name, subset_data, dens,
normalized_density, temp_df, plt)

# Calculate mean scores per cluster
aggregate(. ~ cluster,
          data = FRL_cls,
          FUN = mean)

siz <- data.frame(cls = sort(unique(as.character(paste("Cluster", clusters$cluster)))),
                  frq = as.vector(table(clusters$cluster)))
siz$prp <- siz$frq / sum(siz$frq) * 100

ggplot(siz, aes(x = "", y = frq, fill = cls)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    theme_void() +
    scale_fill_discrete(name = "Clusters",
                        breaks = siz$cls,
                        labels = paste(siz$cls, "-", paste(round(siz$prp, 1), "%", sep =
""")))
```

```
model <- C50::C5.0(as.factor(cluster) ~.,
                   data = FRL_cls)
summary(model)

dta_reg <- dta %>%
  filter(!education.level %in% c("Don't know / not applicable", "Prefer not to say"))

dta_reg$generation.num <- as.numeric(factor(dta_reg$generation,
                                        levels = c("Gen Z","Millennials","Gen X","Boomers")))

dta_reg$education.level.num <- as.numeric(factor(dta_reg$education.level,
                                        levels = c("No formal qualifications", "Secondary edu
cation (e.g. GED/GCSE)", "High school diploma / A-levels", "Technical/community college", "Un
dergraduate degree (BA/BSc/other)","Graduate degree (MA/MSc/MPhil/other)", "Doctorate degree
(PhD/other)")))

apaTables::apa.cor.table(dta_reg[c('generation.num', 'education.level.num')])
```