# additional material

Eyal Grinberg & Yam Rozen

2023-08-22

## Libraries

```r
library(naniar)
library(dplyr)
library(tidyverse)
library(tidytext)
library(stringdist)
library(ggplot2)
```

## Reading Data

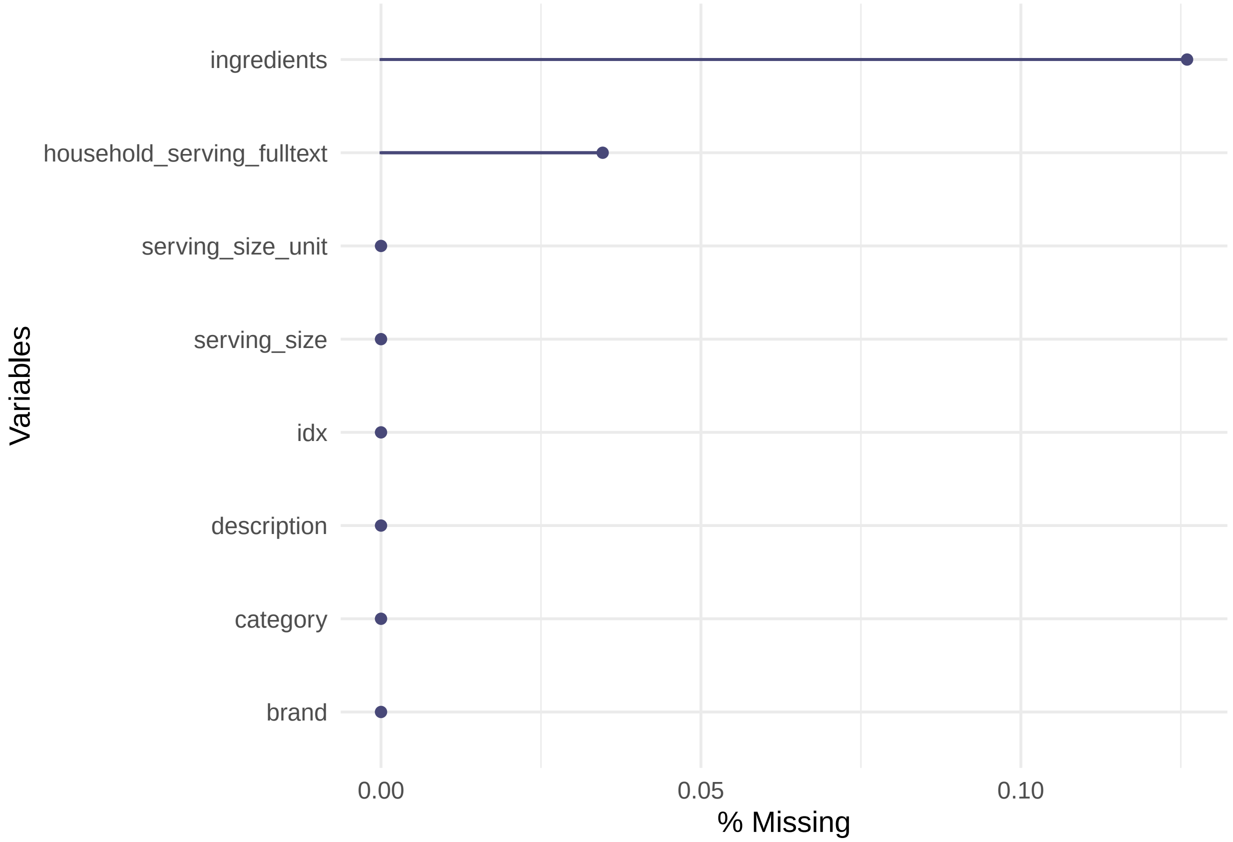
```r
data_food_train <- read_csv("data/food_train.csv")
data_nutrients <- read_csv("data/nutrients.csv")
data_food_nutrients <- read_csv("data/food_nutrients.csv")
data_food_test <- read_csv("data/food_test.csv")
```

## Data Exploratory references

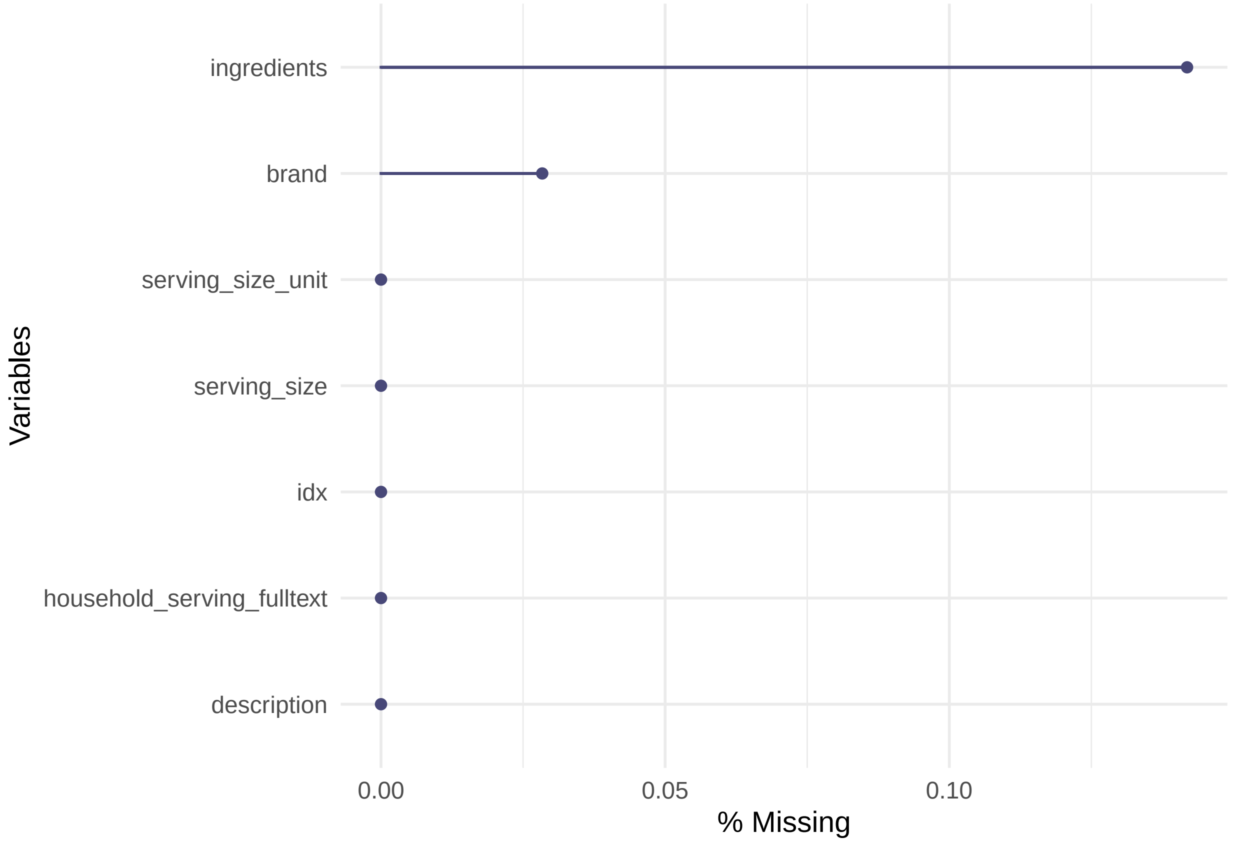
### 1 Data food train

### 1.1 NA values

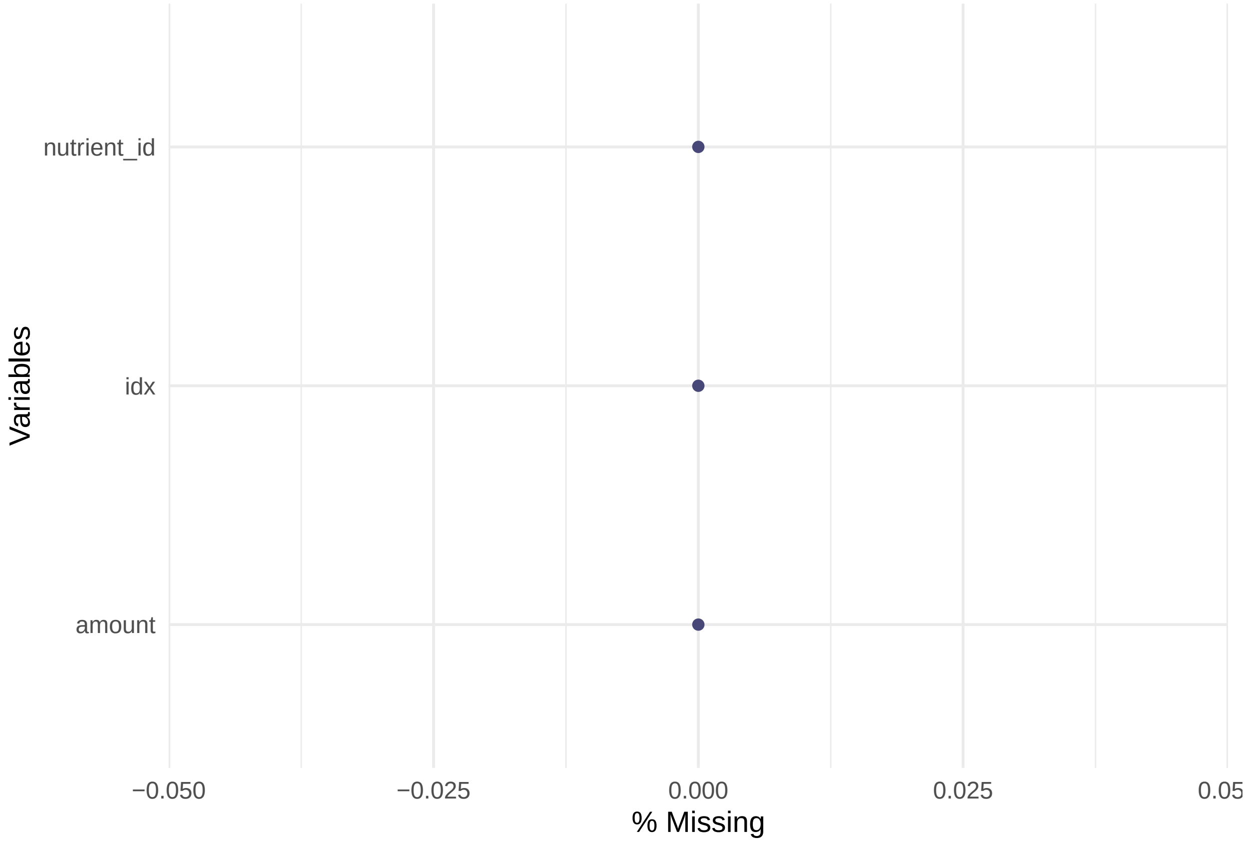
```r
gg_miss_var(data_food_train, show_pct = TRUE)
```



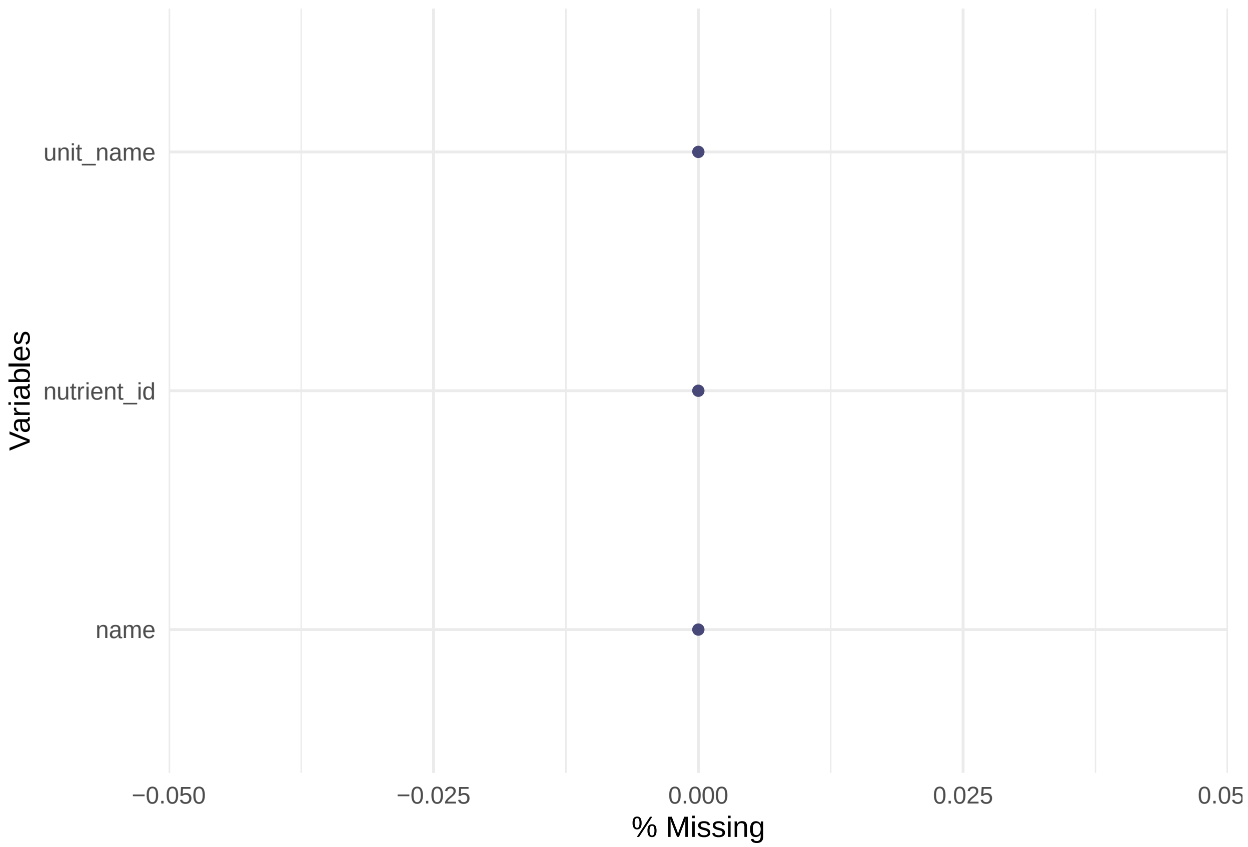
```r
gg_miss_var(data_food_test, show_pct = TRUE)
```



```r
gg_miss_var(data_food_nutrients, show_pct = TRUE)
```



```r
gg_miss_var(data_nutrients, show_pct = TRUE)
```



### 1.2 household_serving_fulltext distance matrix

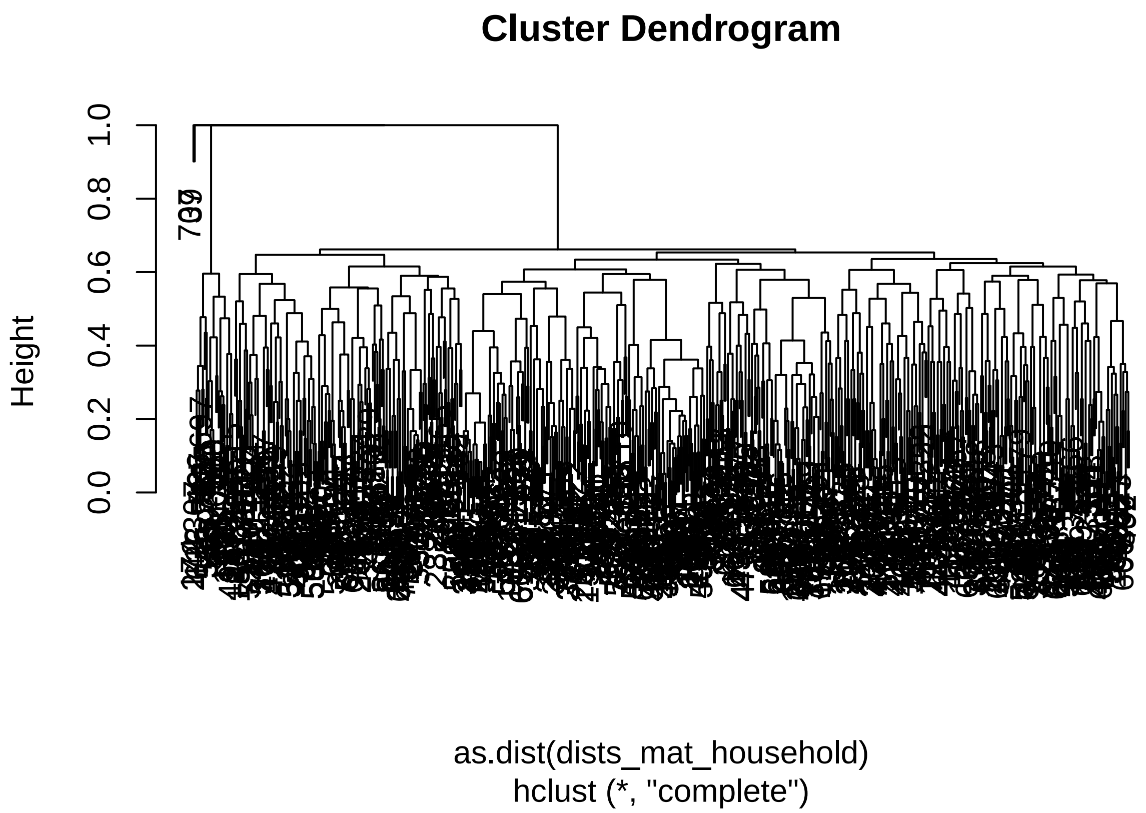
```r
# Let's create a distance matrix for the household_serving_fulltext variable

household_df <- data_food_train %>% select(household_serving_fulltext, category)
household_df$household_serving_only_unit <- gsub("[0-9[:punct:]]", "",
  household_df$household_serving_fulltext) # drop amounts
household_df$household_serving_only_unit <-
  household_df$household_serving_only_unit %>% replace_na("NA")
household_serving_unique <- unique(household_df$household_serving_only_unit)
sum(is.na(household_serving_unique))
```

```
## [1] 0
```

```r
dists_mat_household <- stringdistmatrix(
  household_serving_unique, household_serving_unique, method = "jw")
dists_mat_household[1:10,1:10] # looks pretty good
```

```
##             [,1]       [,2]       [,3]       [,4]      [,5]       [,6]      [,7]
##  [1,] 0.0000000 0.40476190 0.41666667 0.53571429 0.5357143 0.52777778 0.5277778
##  [2,] 0.4047619 0.00000000 0.04166667 0.47619048 0.3809524 0.46031746 0.3571429
##  [3,] 0.4166667 0.04166667 0.00000000 0.32023810 0.2869048 0.40277778 0.2777778
##  [4,] 0.5357143 0.47619048 0.32023810 0.00000000 0.2571429 0.04761905 0.3571429
##  [5,] 0.5357143 0.38095238 0.28690476 0.25714286 0.0000000 0.33730159 0.3571429
##  [6,] 0.5277778 0.46031746 0.40277778 0.04761905 0.3373016 0.00000000 0.4444444
##  [7,] 0.5277778 0.35714286 0.27777778 0.35714286 0.3571429 0.44444444 0.0000000
##  [8,] 0.5000000 0.53571429 0.54166667 0.53571429 0.5357143 0.52777778 0.5277778
##  [9,] 0.5166667 0.32380952 0.34166667 0.32380952 0.4380952 0.30000000 0.4222222
## [10,] 0.5666667 0.40952381 0.37500000 0.49285714 0.3452381 0.56111111 0.4500000
##            [,8]      [,9]     [,10]
##  [1,] 0.5000000 0.5166667 0.5666667
##  [2,] 0.5357143 0.3238095 0.4095238
##  [3,] 0.5416667 0.3416667 0.3750000
##  [4,] 0.5357143 0.3238095 0.4928571
##  [5,] 0.5357143 0.4380952 0.3452381
##  [6,] 0.5277778 0.3000000 0.5611111
##  [7,] 0.5277778 0.4222222 0.4500000
##  [8,] 0.0000000 0.5166667 0.4666667
##  [9,] 0.5166667 0.0000000 0.4166667
## [10,] 0.4666667 0.4166667 0.0000000
```

```
clusters_household <- hclust(as.dist(dists_mat_household))
plot(clusters_household)
```



```
cuts <- cutree(clusters_household, 75)
cuts [1:100]
```

```
##   [1]  1  2  2  3  4  3  5  6  7  8  5  9 10 11  8  9 12 13 14 15 16  4  8  7  7
##  [26]  9 12 17 18 19 20 21 22 12 23  5 14 24 25 26 27 28 29 30 26  1  2  1 14 12
##  [51] 26 26 26  8 26 22 20  3  2 14  1 31 32  5 13 33 34 33 14 35 17 36 37 38 13
##  [76] 20 39  5 40 38 39 14 37 33 30 27 33 41  2 22 21 12 21 22 18 17 15 30 14 42
```

```
# We won't take this idea further.
```

### 1.3 Description Tokenizing

```
# 1 word

data_train_tokenized_description <- data_food_train %>%
  unnest_tokens(word, description) %>% count(category, word, sort = TRUE)
head(data_train_tokenized_description)
```

```
## # A tibble: 6 x 3
##   category                           word          n
##   <chr>                              <chr>     <int>
## 1 cookies_biscuits                   cookies    3152
## 2 chocolate                          chocolate  3142
## 3 candy                              candy      2825
## 4 chips_pretzels_snacks              chips      2516
## 5 chips_pretzels_snacks              potato     1387
## 6 popcorn_peanuts_seeds_related_snacks roasted    1334
```

```
data_train_tokenized_description_grouped_by_cat <- data_train_tokenized_description %>%
  group_by(category) %>% summarise(word, n)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `summarise()` has grouped output by 'category'. You can override using the
## `.groups` argument.
```

```
head(data_train_tokenized_description_grouped_by_cat)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category                   word          n
##   <chr>                      <chr>     <int>
## 1 cakes_cupcakes_snack_cakes cake       1243
## 2 cakes_cupcakes_snack_cakes chocolate   649
## 3 cakes_cupcakes_snack_cakes pie         518
## 4 cakes_cupcakes_snack_cakes cupcakes    368
## 5 cakes_cupcakes_snack_cakes mini        335
## 6 cakes_cupcakes_snack_cakes with        295
```

```
# just chocolate category
data_chocolate <- data_train_tokenized_description_grouped_by_cat[
  data_train_tokenized_description_grouped_by_cat$category == "chocolate" , ]
head(data_chocolate)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category  word          n
##   <chr>     <chr>     <int>
## 1 chocolate chocolate  3142
## 2 chocolate milk       1204
## 3 chocolate dark       1178
## 4 chocolate with        380
## 5 chocolate caramel     338
## 6 chocolate bar         332
```

```
# 2 words

data_train_tokenized_description_2_tokens <- data_food_train %>%
  unnest_tokens(bigram, description, token = "ngrams", n = 2) %>%
  count(category, bigram, sort = TRUE)
head(data_train_tokenized_description_2_tokens)
```

```
## # A tibble: 6 x 3
##   category                            bigram              n
##   <chr>                               <chr>           <int>
## 1 chips_pretzels_snacks               potato chips     1170
## 2 chocolate                           milk chocolate   1042
## 3 chocolate                           dark chocolate   1014
## 4 popcorn_peanuts_seeds_related_snacks trail mix         710
## 5 chips_pretzels_snacks               tortilla chips    609
## 6 cookies_biscuits                    chocolate chip    462
```

```
data_train_tokenized_description_grouped_by_cat_2_tokens <-
  data_train_tokenized_description_2_tokens %>%
  group_by(category) %>% summarise(bigram, n)
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘summarise()‘ has grouped output by ’category’. You can override using the
## ‘.groups‘ argument.
```

```
head(data_train_tokenized_description_grouped_by_cat_2_tokens)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category                bigram              n
##   <chr>                   <chr>           <int>
## 1 cakes_cupcakes_snack_cakes <NA>            185
## 2 cakes_cupcakes_snack_cakes cake with       105
## 3 cakes_cupcakes_snack_cakes chocolate cake    92
## 4 cakes_cupcakes_snack_cakes creme cake        92
## 5 cakes_cupcakes_snack_cakes red velvet        87
## 6 cakes_cupcakes_snack_cakes pound cake        80
```

```r
# just chocolate category
data_chocolate_2_tokens <- data_train_tokenized_description_grouped_by_cat_2_tokens[
  data_train_tokenized_description_grouped_by_cat_2_tokens$category == "chocolate" , ]
head(data_chocolate_2_tokens)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category  bigram                 n
##   <chr>     <chr>              <int>
## 1 chocolate milk chocolate      1042
## 2 chocolate dark chocolate      1014
## 3 chocolate chocolate with       223
## 4 chocolate sea salt             202
## 5 chocolate chocolate bar        179
## 6 chocolate chocolate truffles   139
```

### 1.4 Ingredients Tokenizing

```r
# 1 word

data_train_tokenized_ingredients <- data_food_train %>% unnest_tokens(
  word, ingredients) %>% count(category, word, sort = TRUE)
head(data_train_tokenized_ingredients)
```

```
## # A tibble: 6 x 3
##   category                          word      n
##   <chr>                             <chr> <int>
## 1 cakes_cupcakes_snack_cakes        and   11003
## 2 cakes_cupcakes_snack_cakes        oil   10556
## 3 cookies_biscuits                  flour 10425
## 4 popcorn_peanuts_seeds_related_snacks oil    9788
## 5 cakes_cupcakes_snack_cakes        flour  9737
## 6 cookies_biscuits                  sugar  9640
```

```r
data_train_tokenized_ingredients_grouped_by_cat <-
  data_train_tokenized_ingredients %>% group_by(category) %>% summarise(word, n)
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## ‘summarise()‘ has grouped output by ’category’. You can override using the
## ‘.groups‘ argument.
```

```r
head(data_train_tokenized_ingredients_grouped_by_cat)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category                   word       n
##   <chr>                      <chr>  <int>
## 1 cakes_cupcakes_snack_cakes and    11003
## 2 cakes_cupcakes_snack_cakes oil    10556
## 3 cakes_cupcakes_snack_cakes flour   9737
## 4 cakes_cupcakes_snack_cakes acid    9329
## 5 cakes_cupcakes_snack_cakes sodium  8796
## 6 cakes_cupcakes_snack_cakes sugar   8308
```

```r
# just chocolate category
data_chocolate_ingrediends <- data_train_tokenized_ingredients_grouped_by_cat[
  data_train_tokenized_ingredients_grouped_by_cat$category == "chocolate" , ]
head(data_chocolate_ingrediends)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category  word          n
##   <chr>     <chr>     <int>
## 1 chocolate milk       7243
## 2 chocolate sugar      7072
## 3 chocolate cocoa      5901
## 4 chocolate chocolate  5748
## 5 chocolate butter     5175
## 6 chocolate lecithin   4760
```

```r
# 5 words

data_train_tokenized_ingredients_ngrams_5 <- data_food_train %>%
  unnest_tokens(word, ingredients, token = "ngrams", n = 5) %>%
  count(category, word, sort = TRUE)
head(data_train_tokenized_ingredients_ngrams_5)
```

```
## # A tibble: 6 x 3
##   category                   word                                           n
##   <chr>                      <chr>                                      <int>
## 1 cookies_biscuits           niacin reduced iron thiamine mononitrate    1975
## 2 cookies_biscuits           wheat flour niacin reduced iron             1923
## 3 cookies_biscuits           flour niacin reduced iron thiamine          1893
## 4 cakes_cupcakes_snack_cakes thiamine mononitrate riboflavin folic acid  1875
## 5 cookies_biscuits           thiamine mononitrate riboflavin folic acid  1792
## 6 cakes_cupcakes_snack_cakes iron thiamine mononitrate riboflavin folic  1776
```

```r
data_train_tokenized_ingredients_grouped_by_cat_ngrams_5 <-
  data_train_tokenized_ingredients_ngrams_5 %>%
  group_by(category) %>% summarise(word, n)
```

```
## Warning: Returning more (or less) than 1 row per ‘summarise()‘ group was deprecated in
## dplyr 1.1.0.
## i Please use ‘reframe()‘ instead.
## i When switching from ‘summarise()‘ to ‘reframe()‘, remember that ‘reframe()‘
##   always returns an ungrouped data frame and adjust accordingly.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## ‘summarise()‘ has grouped output by ’category’. You can override using the
## ‘.groups‘ argument.
```

```
head(data_train_tokenized_ingredients_grouped_by_cat_ngrams_5)
```

```
## # A tibble: 6 x 3
## # Groups:   category [1]
##   category                  word                                              n
##   <chr>                     <chr>                                         <int>
## 1 cakes_cupcakes_snack_cakes thiamine mononitrate riboflavin folic acid    1875
## 2 cakes_cupcakes_snack_cakes iron thiamine mononitrate riboflavin folic    1776
## 3 cakes_cupcakes_snack_cakes niacin reduced iron thiamine mononitrate      1510
## 4 cakes_cupcakes_snack_cakes reduced iron thiamine mononitrate riboflavin  1481
## 5 cakes_cupcakes_snack_cakes flour niacin reduced iron thiamine            1447
## 6 cakes_cupcakes_snack_cakes wheat flour niacin reduced iron               1273
```

```
# Not very helpful
```

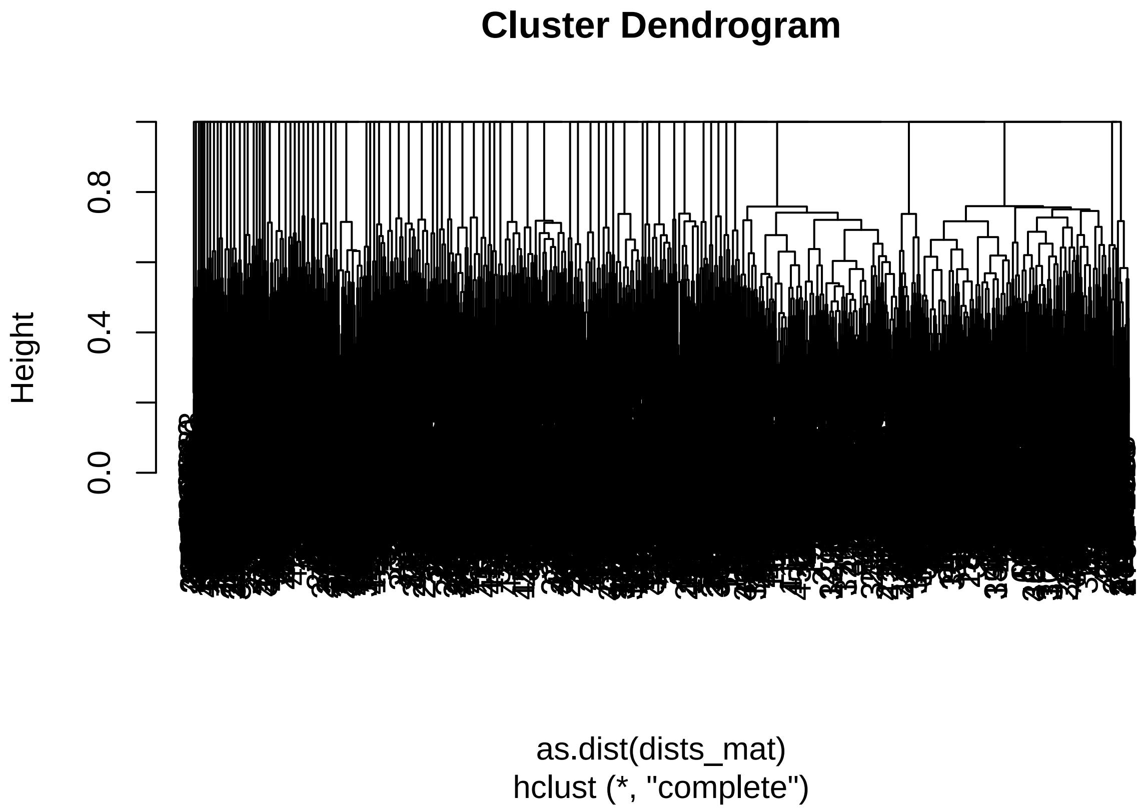
### 1.5 Brand distance matrix

```
brands_unique <- unique(data_food_train$brand)
length(brands_unique) # 4783
```

```
## [1] 4783
```

```
# We can see that there are many different brands, is it possible to merge some of them?
```

```
# Let's create a distance matrix between the brands
dists_mat <- stringdistmatrix(brands_unique, brands_unique, method = "jw")
clusters <- hclust(as.dist(dists_mat))
plot(clusters)
```



```r
cuts <- cutree(clusters, 3000)
cuts[1:100]
```

```
##   [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
##  [51] 51 52 53 54 55 56 57 58 59 60 61 62 63 64 26 65 66 67 68 69 70 71 72 73 74
##  [76] 75 76 77 78 79 80 81 82 83 23 84 85 86 87 88 89  7 22 90 91 92 93 94 95 96
```

```r
# after watching the results we saw that brands 26 and 65 were clustered to the same cluster.
brands_unique[26]
```

```
## [1] "inventure foods, inc."
```

```r
brands_unique[65]
```

```
## [1] "interbake foods inc."
```

```r
# after manually scanning the results we noticed that many brands
# have the following prefixes: inc, ltd, llc, co, corp, company
num_inc <- sum(str_detect(brands_unique, "inc"))
num_ltd <- sum(str_detect(brands_unique, "ltd"))
num_llc <- sum(str_detect(brands_unique, "llc"))
num_co <- sum(str_detect(brands_unique, "co"))
```

```
num_corp <- sum(str_detect(brands_unique, "corp"))
num_company <- sum(str_detect(brands_unique, "company"))
num_inc
```

```
## [1] 945
```

```
num_ltd
```

```
## [1] 78
```

```
num_llc
```

```
## [1] 434
```

```
num_co
```

```
## [1] 1075
```

```
num_corp
```

```
## [1] 136
```

```
num_company
```

```
## [1] 231
```

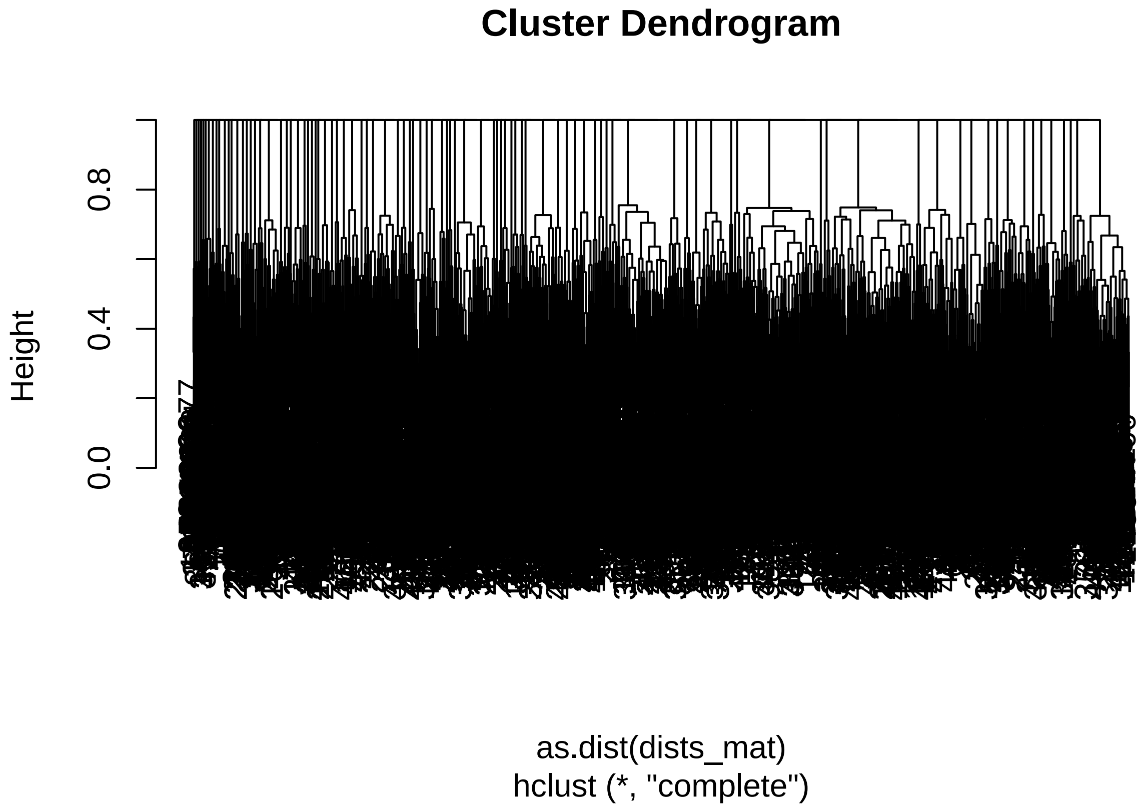
```
# These prefixes make the similarity algorithm find similarities that
# we don't really want, Let's drop them.

words_to_drop <- c(" inc", " ltd", " llc", " co ", "corp", " company")
pattern <- paste(words_to_drop, collapse = "|")
filtered_unique_brands <- gsub(pattern, "", brands_unique)
filtered_unique_brands <- gsub("\\s{2,}", " ", filtered_unique_brands)
length(unique(filtered_unique_brands)) # 4745
```

```
## [1] 4745
```

```
# Not a big difference

# Let's create a distance matrix between the filtered brands
dists_mat <- stringdistmatrix(filtered_unique_brands, filtered_unique_brands, method = "jw")
dists_mat[1:10,1:10]
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] 0.0000000 0.4860806 0.4980159 0.4761905 0.4285714 0.5293651 0.5056022
##  [2,] 0.4860806 0.0000000 0.4962607 0.4168956 0.4860806 0.5709402 0.5340623
##  [3,] 0.4980159 0.4962607 0.0000000 0.4900794 0.5456349 0.6527778 0.4948257
##  [4,] 0.4761905 0.4168956 0.4900794 0.0000000 0.4285714 0.5031746 0.4579832
##  [5,] 0.4285714 0.4860806 0.5456349 0.4285714 0.0000000 0.5658730 0.4026611
##  [6,] 0.5293651 0.5709402 0.6527778 0.5031746 0.5658730 0.0000000 0.4156863
##  [7,] 0.5056022 0.5340623 0.4948257 0.4579832 0.4026611 0.4156863 0.0000000
##  [8,] 0.4940476 0.4405271 0.4325397 0.4384921 0.4755291 0.5091270 0.3167600
##  [9,] 0.5367965 0.5608003 0.5801768 0.5995671 0.4641655 0.6656566 0.5019806
## [10,] 0.5938645 0.4145299 0.5961538 0.4688645 0.5088523 0.4752137 0.4213532
##            [,8]      [,9]     [,10]
##  [1,] 0.4940476 0.5367965 0.5938645
##  [2,] 0.4405271 0.5608003 0.4145299
##  [3,] 0.4325397 0.5801768 0.5961538
##  [4,] 0.4384921 0.5995671 0.4688645
##  [5,] 0.4755291 0.4641655 0.5088523
##  [6,] 0.5091270 0.6656566 0.4752137
##  [7,] 0.3167600 0.5019806 0.4213532
##  [8,] 0.0000000 0.4835859 0.4047009
##  [9,] 0.4835859 0.0000000 0.4763570
## [10,] 0.4047009 0.4763570 0.0000000
```

```
clusters <- hclust(as.dist(dists_mat))
plot(clusters)
```



```
cuts <- cutree(clusters, 3000)
cuts[1:100]
```

```
##   [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 31 49
##  [51] 50 51 52 53 54 55 56 57 58 59 60 61 62 63 26 64 65 66 67 68 69 70 71 72 73
##  [76] 74 75 76 77 78 79 80 81 82 23 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97
```

# 2 data_nutrients + data_food_nutrients

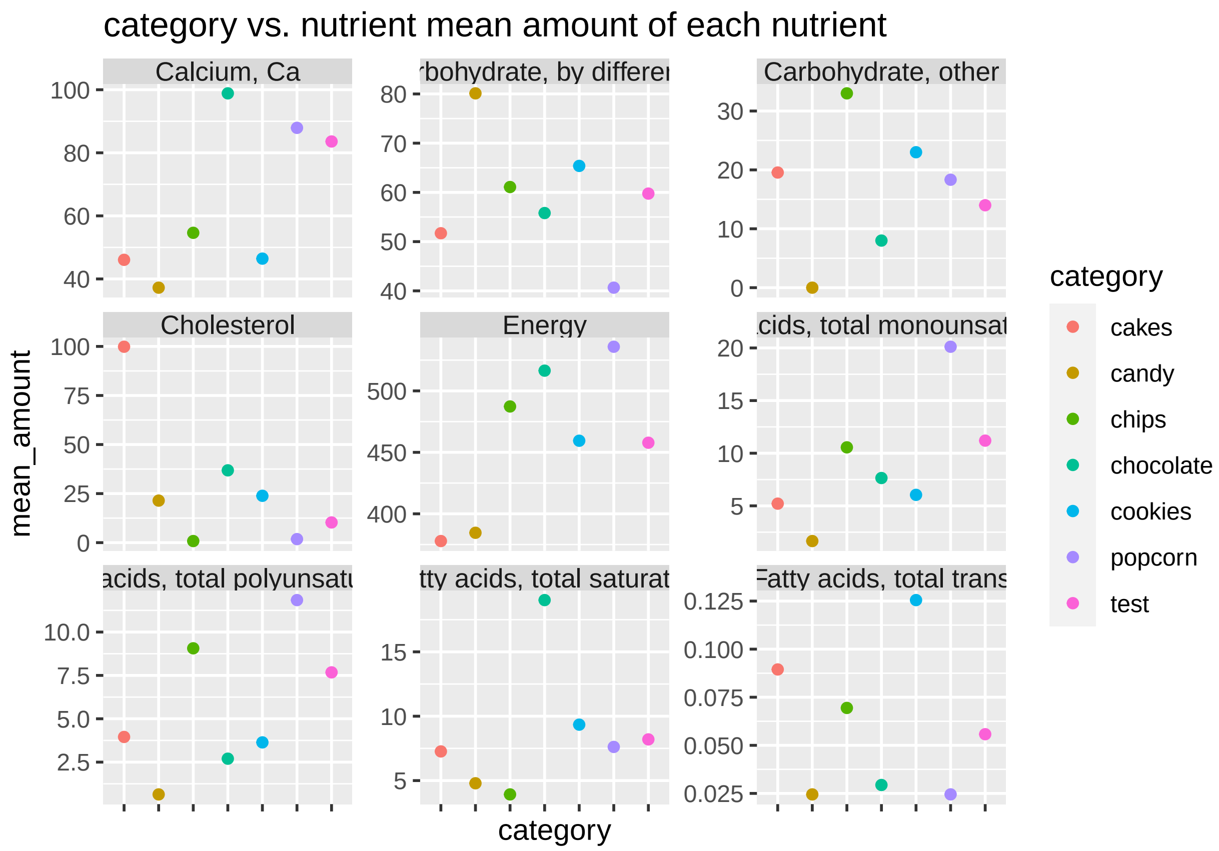
## 2.1 Plots mean amount of each nutrient by category

```
merged_df_nutrients <- merge(data_food_nutrients, data_nutrients,
                             by = "nutrient_id", all.x = TRUE) %>% arrange(idx)

merged_df_nutrients <- merged_df_nutrients[-1]

data_food_test$category <- "unknown"
data_food <- rbind(data_food_train, data_food_test)
merged_df_nutrients <- merge(merged_df_nutrients, data_food[, c(1,8)],
                             by = "idx", all.x = TRUE)

df_nutrients_mean_by_cat <- merged_df_nutrients %>%
  group_by(category, name) %>% mutate(mean_amount = mean(amount))

nuts_splitted_by_cat <- split(merged_df_nutrients , merged_df_nutrients$category)

cakes_mean_by_nut <- nuts_splitted_by_cat$cakes_cupcakes_snack_cakes %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
choco_mean_by_nut <- nuts_splitted_by_cat$chocolate %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
popcorn_mean_by_nut <- nuts_splitted_by_cat$popcorn_peanuts_seeds_related_snacks %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
candy_mean_by_nut <- nuts_splitted_by_cat$candy %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
chips_mean_by_nut <- nuts_splitted_by_cat$chips_pretzels_snacks %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
cookies_mean_by_nut <- nuts_splitted_by_cat$cookies_biscuits %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
test_mean_by_nut <- nuts_splitted_by_cat$unknown %>%
  group_by(name) %>% reframe(mean_amount = mean(amount))
head(cakes_mean_by_nut)
```

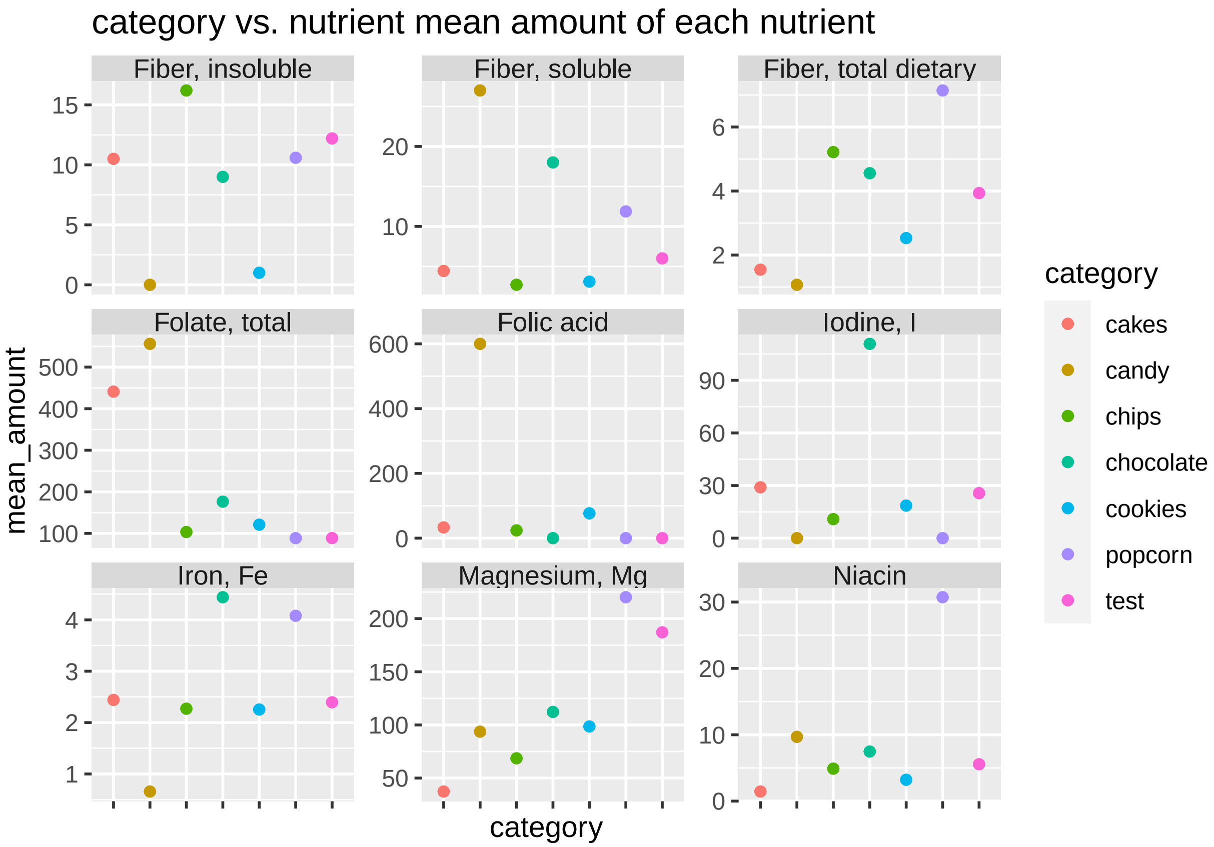
```
## # A tibble: 6 x 2
##   name                                mean_amount
##   <chr>                                     <dbl>
## 1 Calcium, Ca                               46.1
## 2 Carbohydrate, by difference               51.7
## 3 Carbohydrate, other                       19.6
## 4 Cholesterol                               99.8
## 5 Energy                                   378.
## 6 Fatty acids, total monounsaturated         5.21
```

```
df_list <- list(cakes_mean_by_nut, candy_mean_by_nut, popcorn_mean_by_nut,
  choco_mean_by_nut, chips_mean_by_nut, cookies_mean_by_nut, test_mean_by_nut)

nutrients_mean_amount_with_zero_amounts <- df_list %>% reduce(full_join, by = "name")
colnames(nutrients_mean_amount_with_zero_amounts) <-
```

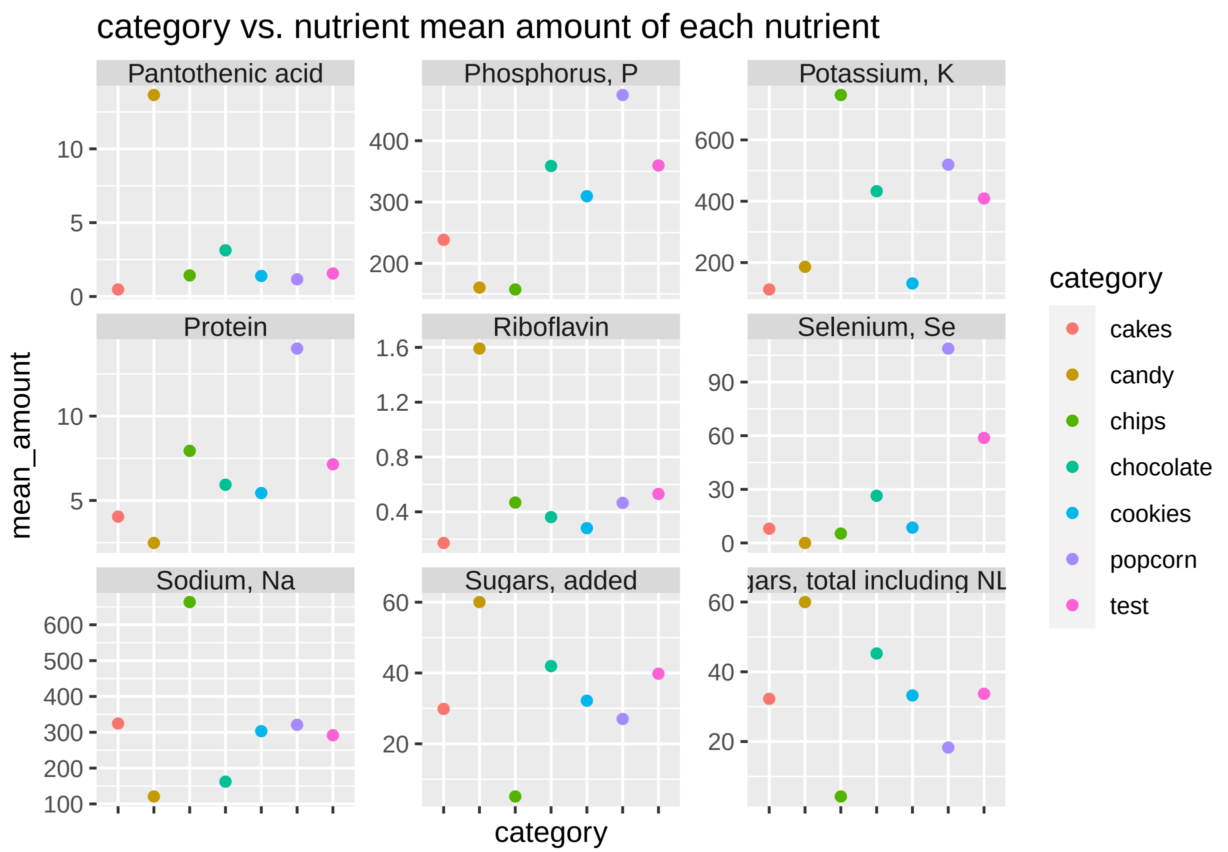
```
  c("nutrient", "cakes", "candy", "popcorn", "chocolate", "chips", "cookies", "test")

# replace NAs with 0
nutrients_mean_amount_with_zero_amounts[is.na(
  nutrients_mean_amount_with_zero_amounts)] <- 0

# Visualization

ggplot_nutrients <- function(pivoted_df, start, end) {
  ggplot(data = pivoted_df[start:end,], mapping = aes(x = category, y = mean_amount,
  color = category)) + geom_point() + facet_wrap(. ~ nutrient, scales = "free_y") +
  labs(title = "category vs. nutrient mean amount of each nutrient",
  x = "category", y ="mean_amount") + theme(strip.text.x = element_text(
  size = 10, margin = margin()),axis.text.x.bottom = element_blank(),
  strip.text.y = element_text(size = 20, margin = margin()))
}

nuts_pivoted <- nutrients_mean_amount_with_zero_amounts %>%
  pivot_longer(!nutrient, names_to = "category", values_to = "mean_amount")

ggplot_nutrients(nuts_pivoted, 1, 63)
```



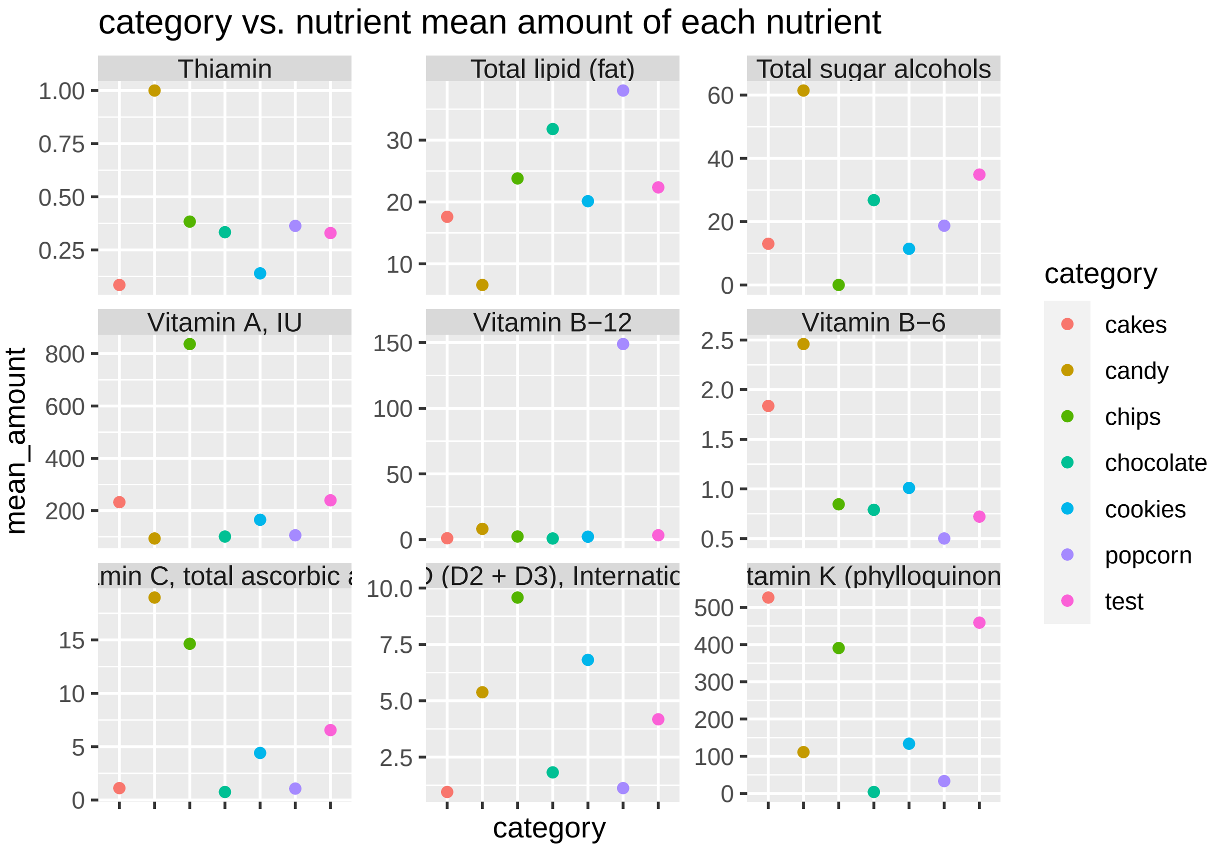
```
ggplot_nutrients(nuts_pivoted, 64, 126)
```



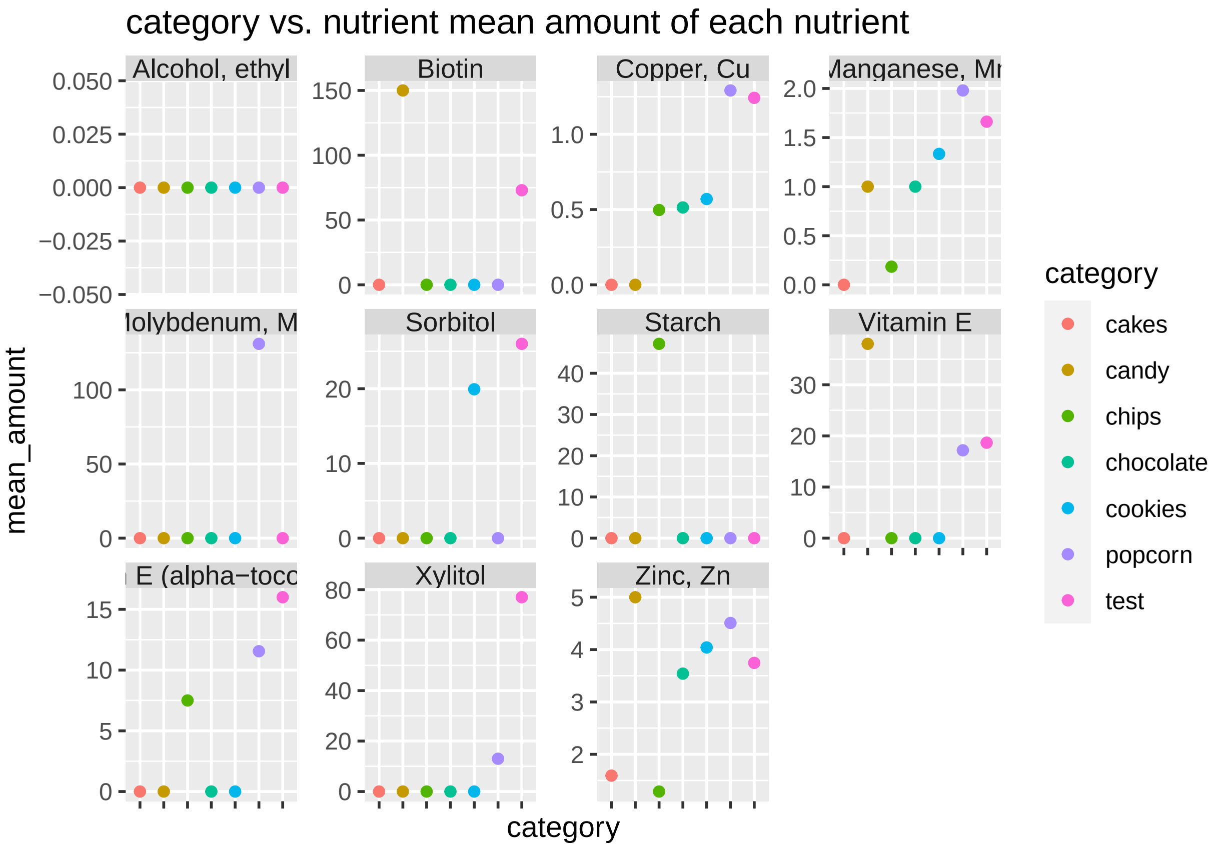
```
ggplot_nutrients(nuts_pivoted, 127, 189)
```



```r
ggplot_nutrients(nuts_pivoted, 190, 252)
```



```
ggplot_nutrients(nuts_pivoted, 253, 329)
```


category vs. nutrient mean amount of each nutrient
Alcohol, ethyl
Biotin
Copper, Cu
Manganese, Mn
Molybdenum, M
Sorbitol
Starch
Vitamin E
E (alpha−toco
Xylitol
Zinc, Zn
mean_amount
category
0.050
0.025
0.000
−0.025
−0.050
150
100
50
0
1.0
0.5
0.0
2.0
1.5
1.0
0.5
0.0
100
50
0
20
10
0
40
30
20
10
0
30
20
10
0
15
10
5
0
80
60
40
20
0
5
4
3
2
category
cakes
candy
chips
chocolate
cookies
popcorn
test