Grand Valley Magazine Project

Matthew Dickinson & Foster Thorburn

2023-11-09

Data Import

```
data_import <- read_csv("STA 419 GV Magazine Survey.csv")
```

```
## Rows: 643 Columns: 65
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (65): ResponseID, ResponseSet, IPAddress, StartDate, EndDate, RecipientL...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data_dict <- read_csv("Data Dictionary for GVM survey - Final.csv")
```

```
## Rows: 51 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): question_number, question_text, variable_name, variable_type, varia...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
gvm_main <- data_import |>
  select(12:62) |>
  slice(c(-1,-2))

colnames(gvm_main) <- data_dict$variable_name
```

Cleaning

- columns (likert) to numeric
- where_read to categorize
- num issues: 1,2,6,7 (0, 1, 2, 3)
- gvsu_engagment to category
- age to be actual categorize
- num_articles change 'NOT_APPLICABLE' to "more than 10"
- drive to website category
- website_only to yes/no
- website_engagment to category

```
gvm_clean <- gvm_main |>
  select(-1, -51) |>
  mutate(
```

```
    where_read = case_when(
      where_read == "1" ~ "GVM Website",
      where_read == "2" ~ "Print issues",
      where_read == "3" ~ "Both",
      TRUE ~ "I don't read the GVM"),
    num_issues = as.numeric(case_when(
      num_issues == "1" ~ 0,
      num_issues == "2" ~ 2,
      num_issues == "6" ~ 3,
      TRUE ~ 1)),
    gvsu_engagement = case_when(
      gvsu_engagement == "1" ~ "Disagree",
      gvsu_engagement == "2" ~ "Neutral",
      TRUE ~ "Agree"),
    age = case_when(
      age == "1" ~ "17-24",
      age == "2" ~ "25-35",
      age == "3" ~ "36-49",
      age == "4" ~ "50-65",
      TRUE ~ "66+"
    ))
```

```
gvm_clean$num_articles <- as.numeric(gvm_clean$num_articles)
```

```
## Warning: NAs introduced by coercion
```

```
#gvm_clean <- gvm_clean %>% mutate(num_articles=recode(num_articles,
#                                  "NA" = 15))
gvm_clean$num_articles[is.na(gvm_clean$num_articles)] <- 15

gvm_clean <- gvm_clean %>%
  mutate(website_only = case_when(
    website_only == '1' ~ "Yes",
    website_only == '2' ~ "I don't know/I'm not sure",
    website_only == '3' ~ "No",
    TRUE ~ as.character(website_only)
  ))

gvm_clean <- gvm_clean %>%
  mutate(website_engagment = case_when(
    website_engagment == '1' ~ "Larger variety of content",
    website_engagment == '2' ~ "Easier navigation",
    website_engagment == '3' ~ "It is currently engaging",
    website_engagment == '8' ~ "More specific recommendations",
    website_engagment == '4' ~ "I don't know/I'm not sure",
    website_engagment == '5' ~ "I don't use the website",
    website_engagment == '7' ~ "Other",
    TRUE ~ as.character(website_engagment)
  ))


gvm_clean <- gvm_clean %>%
  mutate(drive_to_website = case_when(
    drive_to_website == '1' ~ 'Word of Mouth',
```

```
    drive_to_website == '2' ~ 'Email',
    drive_to_website == '3' ~ 'Social Media',
    drive_to_website == '4' ~ 'Link within printed Issue',
    drive_to_website == '5' ~ 'Doing research',
    TRUE ~ as.character(drive_to_website)
  ))

gvm_clean <- gvm_clean %>%
  mutate(opting = case_when(
    opting == '1' ~ 'Yes',
    opting == '2' ~ 'No',
    TRUE ~ as.character(opting)
  ))

gvm_clean <- type.convert(gvm_clean, as.is = TRUE)
```

```
# vector for column names
relation_cols <- gvm_clean |> select(starts_with("relation")) |> colnames()

# empty vector to store counts
relation_counts <- vector()

# finding counts for each column name
for (i in 1:length(relation_cols)){
  # print(gvm_clean[relation_cols[i]])
  relation_counts[i] = length(which(gvm_clean[relation_cols[i]] == 1))
}

# storing information in df
relation_df <- data.frame(
  relation_cols,
  relation_counts
)

# cleaning names
relation_df <- relation_df |> mutate(relation_cols = str_replace_all(relation_cols, "relation_", ""))
```

```
#Most Info df
counts <-c(
length(which(gvm_clean$most_info_gv_emails == 1)),
length(which(gvm_clean$most_info_gvm_print == 1)),
length(which(gvm_clean$most_info_gvm_website == 1)),
length(which(gvm_clean$most_info_gv_publications == 1)),
length(which(gvm_clean$most_info_media == 1)),
length(which(gvm_clean$most_info_wordofmouth_alumni == 1)),
length(which(gvm_clean$most_info_lanthorn == 1)),
length(which(gvm_clean$most_info_socialmedia == 1)),
length(which(gvm_clean$most_info_other == 1))
)

most_info_df <- data.frame(
  answer = c("Emails from GVSU","Grand Valley Magazine Print Issues","Grand Valley Magazine Website","O
  count = counts
)
```

```r
# vector for column names
action_cols <- gvm_clean |> select(starts_with("action")) |> colnames()

# empty vector to store counts
action_counts <- vector()

# finding counts for each column name
for (i in 1:length(action_cols)){
  # print(gvm_clean[relation_cols[i]])
  action_counts[i] = length(which(gvm_clean[action_cols[i]] == 1))
}


# storing information in df
action_df <- data.frame(
  action_cols,
  action_counts
)

# cleaning names
action_df <- action_df |> mutate(action_cols = str_replace_all(action_cols, "action_", ""))
```

Creating new variables

```r
gvm_clean$avid_reader <- as.factor(gvm_clean$num_articles)

gvm_clean <- gvm_clean %>%
  mutate(avid_reader = case_when(
    avid_reader %in% c('0', '1', '2', '3', '4') ~ '0-4 Articles',
    avid_reader %in% c('5', '6', '7', '8', '9', '10', '15') ~ '5-10+',
    TRUE ~ as.character(avid_reader)
  ))

table(gvm_clean$avid_reader)
```

```
##
## 0-4 Articles        5-10+
##          454          187
```

```r
gvm_clean <- gvm_clean |>
  mutate(
    key_pop = as.factor(case_when(
      (relation_alumni == 1 & age %in% c("36-49", "50-65", "66+")) ~ "Alumni over 35",
      TRUE ~ "Other")))
```

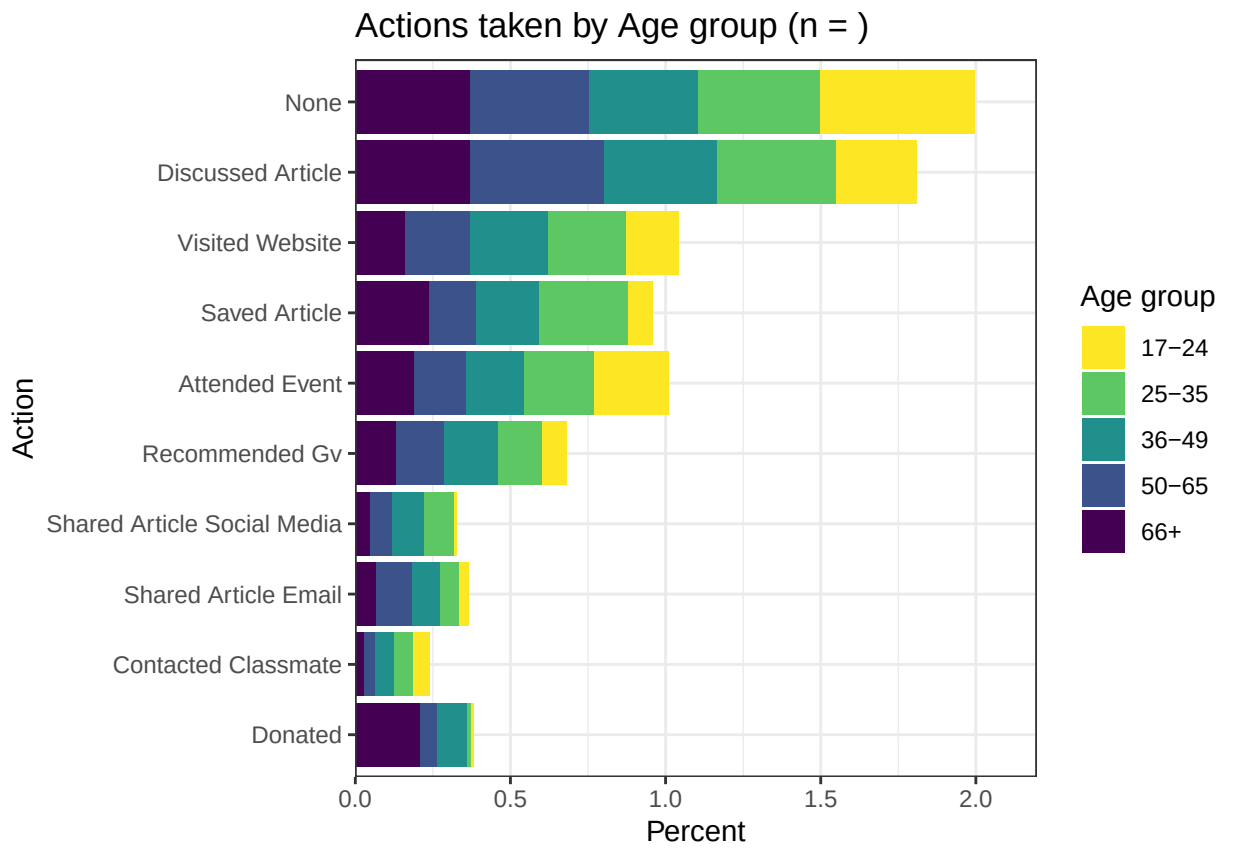
```r
gvm_clean <- gvm_clean |>
  mutate(num_actions = rowSums(gvm_clean |> select(starts_with("action_")), na.rm = TRUE)
    )
```

Visualizations

```r
# make age a factor because it will make using it as a variable easier
gvm_clean$age <- as.factor(gvm_clean$age)

# create new df for the data we want
age_by_action <- gvm_clean |>
  select(age, starts_with("action")) |> # grabbing the variables of interest
  pivot_longer(!age, names_to = "action", values_to = "count") |> # pivoting
  group_by(age, action) |> # grouping to make summarizing
  summarize(count = sum(count, na.rm = TRUE),
            n = n()) |> # getting the counts for each category
  mutate(action = str_replace_all(action, "action_", ""), # simple text cleaning
         action = str_replace_all(action, "_", " "),
         action = str_to_title(action),
         percent2 = (count/n))
```

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

```r
age_by_action |>
  ggplot(aes(fct_reorder(action, (percent2)), percent2, fill = age)) + # grab variables and reorder act
  geom_bar(stat = "identity")  +
  theme_bw() +
  # theme(axis.text.x = element_text(angle = 45, vjust = .6)) +
  labs(
    x = "Action",
    y = "Percent",
    fill = "Age group",
    title = "Actions taken by Age group (n = )"
  ) +
  scale_fill_viridis_d(direction = -1) +
  scale_y_continuous(expand = expansion(mult = c(0, 0.1))) + # this line makes sure there is no gap bet
  coord_flip()
```




```
# compare within groups
# focus
```

```
gvm_clean <- rename(gvm_clean,'info_text'='most_info_text')
gvm_clean$age <- as.factor(gvm_clean$age)
# create new df for the data we want
age_by_most_info <- gvm_clean |>
  select(age, starts_with("most_info")) |> # grabbing the variables of interest
  pivot_longer(!age, names_to = "most_info", values_to = "count") |> # pivoting
  group_by(age, most_info) |> # grouping to make summarizing
  summarize(n = sum(count, na.rm = TRUE)) |> # getting the counts for each category
  mutate(most_info = str_replace_all(most_info, "most_info_", ""), # simple text cleaning
         most_info = str_replace_all(most_info, "_", " "),
         most_info = str_to_title(most_info))
```

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

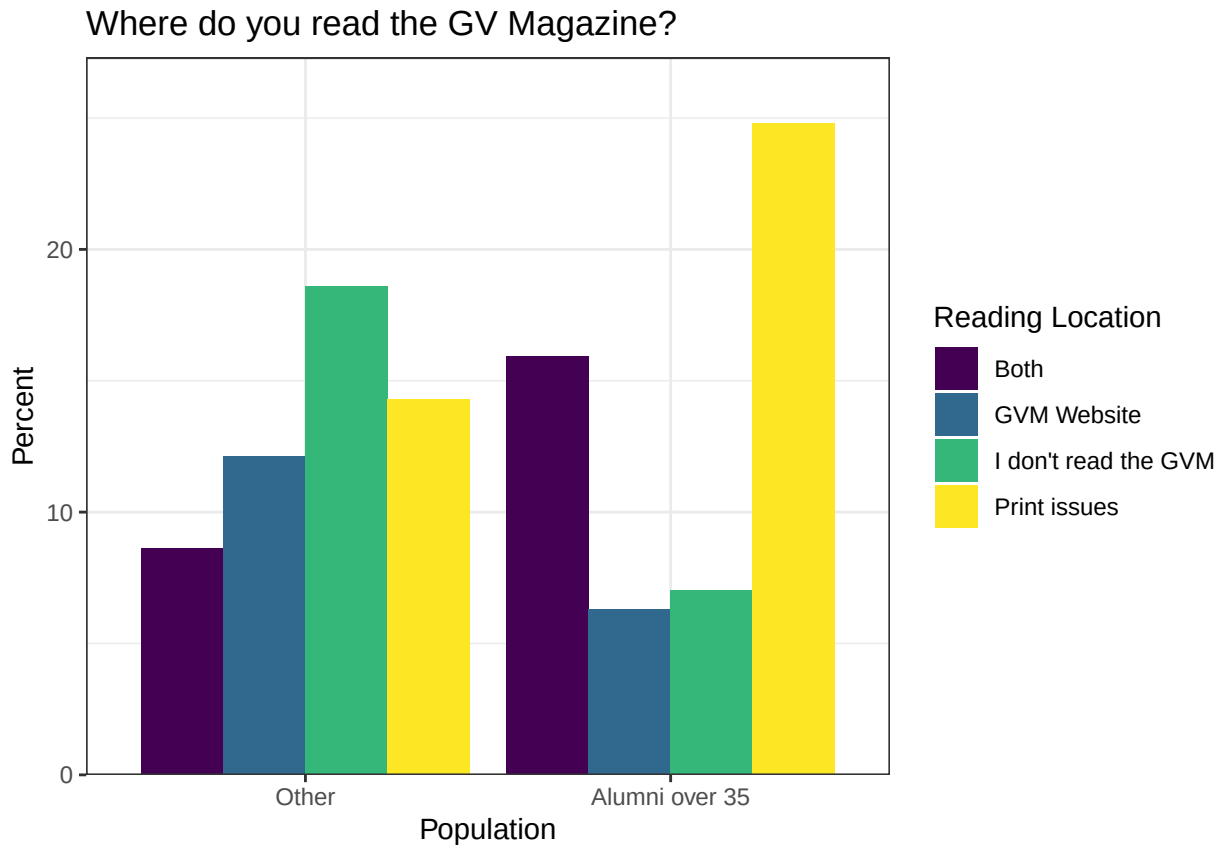
```
key_engage_read <- gvm_clean |>
  group_by(key_pop, gvsu_engagement, where_read) |>
  summarize(n = n(),
            percent = case_when(
              key_pop == "Alumni over 35" ~ n/nrow(gvm_clean |> filter(key_pop == "Alumni over 35")),
              TRUE ~ n/nrow(gvm_clean |> filter(key_pop == "Other"))
            ) * 100)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
```

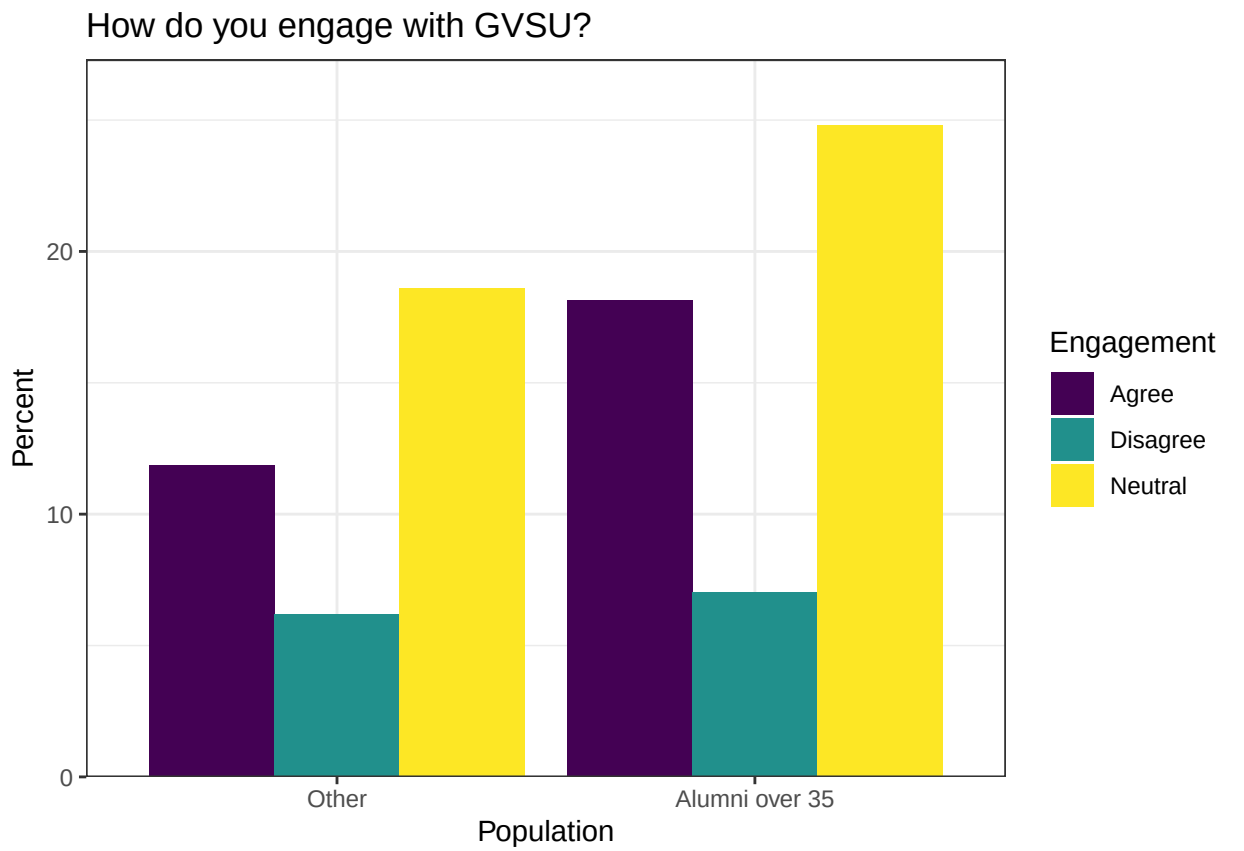
```
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `summarise()` has grouped output by 'key_pop', 'gvsu_engagement', 'where_read'.
## You can override using the `.groups` argument.
```

```
key_engage_read |>
  ggplot(aes(fct_reorder(key_pop, (percent)), percent, fill = where_read)) +
  geom_col(position = "dodge") +
  theme_bw() +
  labs(x = "Population", y = "Percent", fill = "Reading Location", title = "Where do you read the GV Mag
  scale_y_continuous(expand = expansion(mult = c(0, 0.1))) +
  scale_fill_viridis_d()
```




```
key_engage_read |>
  ggplot(aes(fct_reorder(key_pop, (percent)), percent, fill = gvsu_engagement)) +
  geom_col(position = "dodge") +
  theme_bw() +
  labs(x = "Population", y = "Percent", fill = "Engagement", title = "How do you engage with GVSU?") +
  scale_y_continuous(expand = expansion(mult = c(0, 0.1))) +
  scale_fill_viridis_d()
```

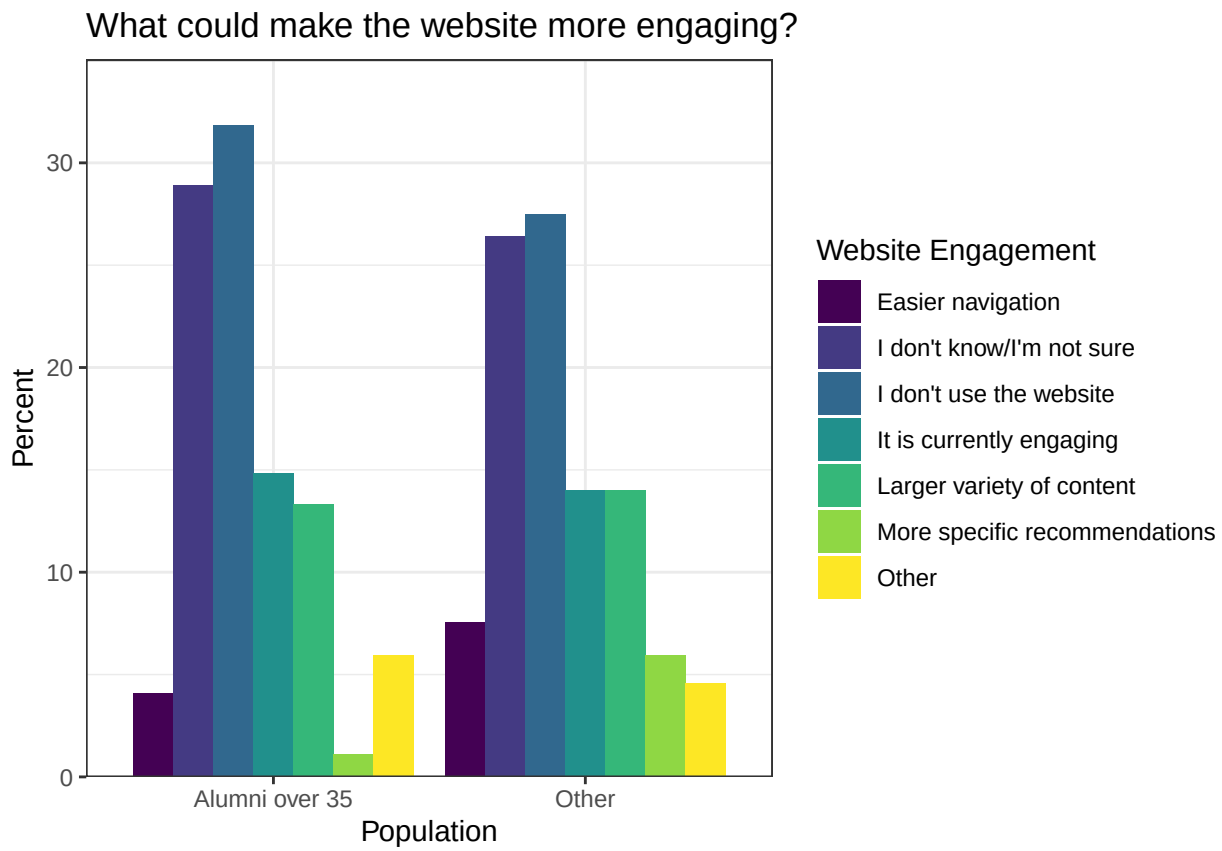



```
# age on website engagement
key_web_engage <- gvm_clean |>
  group_by(key_pop, website_engagment) |>
  summarize(n = n(),
            percent = case_when(
              key_pop == "Alumni over 35" ~ n/nrow(gvm_clean |> filter(key_pop == "Alumni over 35")),
              TRUE ~ n/nrow(gvm_clean |> filter(key_pop == "Other")
            ))*100)
```

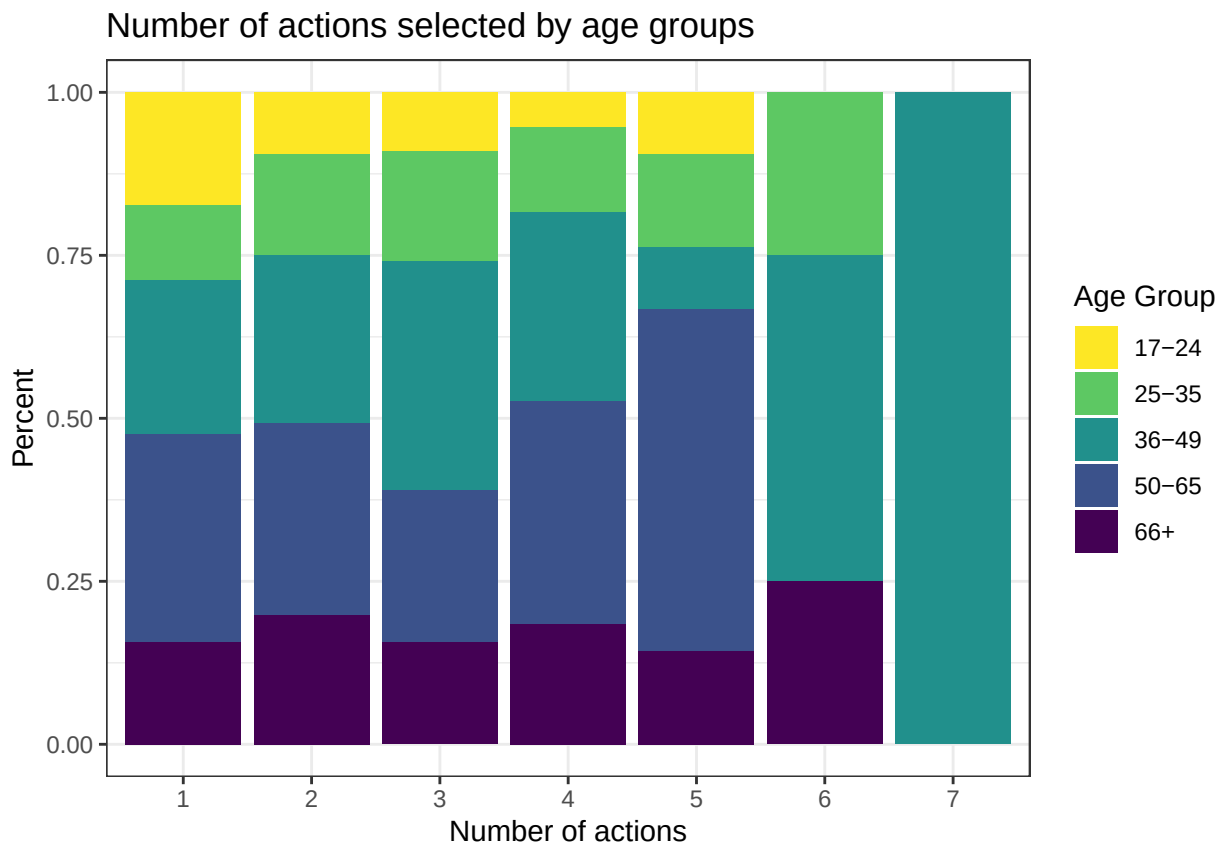
```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call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

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

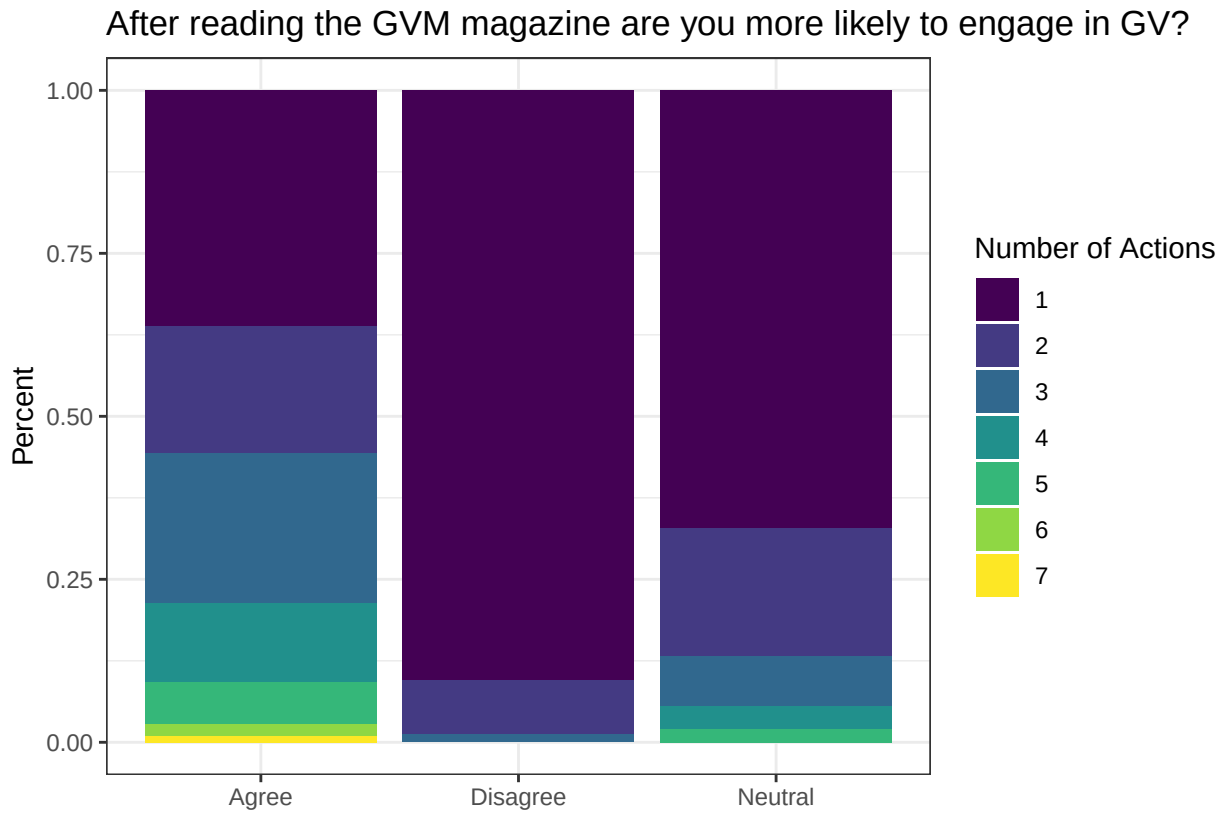
```
key_web_engage |>
  ggplot(aes(key_pop, percent, fill = as.factor(website_engagment))) +
  geom_col(position = "dodge") +
  theme_bw() +
  labs(x = "Population", y = "Percent", fill = "Website Engagement", title = "What could make the websi
  scale_y_continuous(expand = expansion(mult = c(0, 0.1))) +
  scale_fill_viridis_d()
```




```
gvm_clean |>
  ggplot(aes(as.factor(num_actions), fill = age)) +
  geom_bar(position = "fill") +
  theme_bw() +
  labs(
    x = "Number of actions",
    y = "Percent",
    fill = "Age Group",
    title = "Number of actions selected by age groups"
  ) +
  scale_fill_viridis_d(direction = -1)
```




```
# find average number of actions would be better
gvm_clean |>
  ggplot(aes(as.factor(gvsu_engagement), fill = as.factor(num_actions))) +
  geom_bar(position = "fill") +
  theme_bw() +
  labs(
    x = "",
    y = "Percent",
    fill = "Number of Actions",
    title = "After reading the GVM magazine are you more likely to engage in GV?"
  ) +
  scale_fill_viridis_d()
```

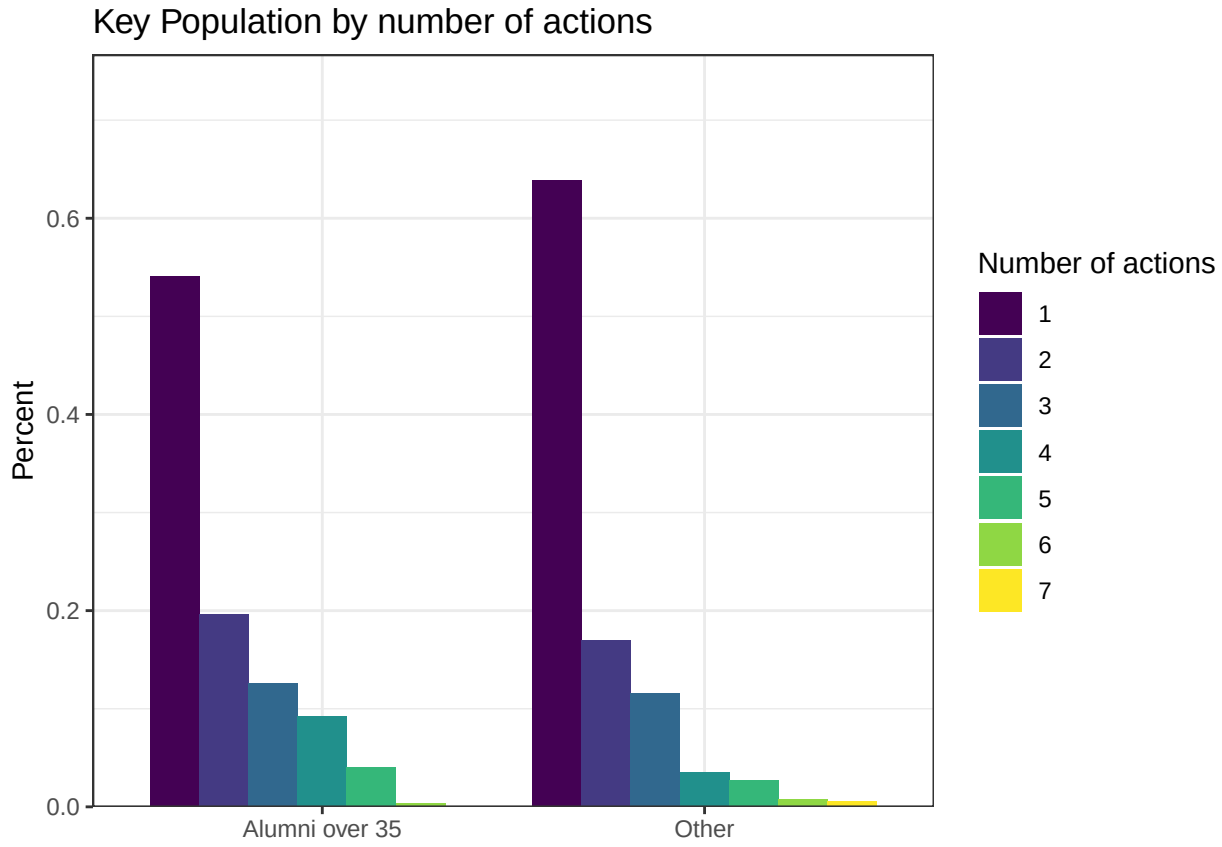



```
num_per_pop <- gvm_clean |>
  select(age, key_pop, num_actions) |>
  group_by(key_pop) |>
  summarize(n = n()) |>
  pull(n)

actions_key_pop <- gvm_clean |>
  select(key_pop, num_actions) |>
  group_by(key_pop, num_actions) |>
  summarize(n = n()) |>
  ungroup() |>
  add_row(num_actions = 7, key_pop = "Alumni over 35", n = 0) |>
  arrange(key_pop, num_actions) |>
  mutate(total = sort(rep(num_per_pop, 7), decreasing = FALSE),
         total2 = case_when(
           key_pop == "Other" ~ num_per_pop[2],
           TRUE ~ num_per_pop[1]),
         percent = n / total2)|>
  arrange(desc(percent))
```

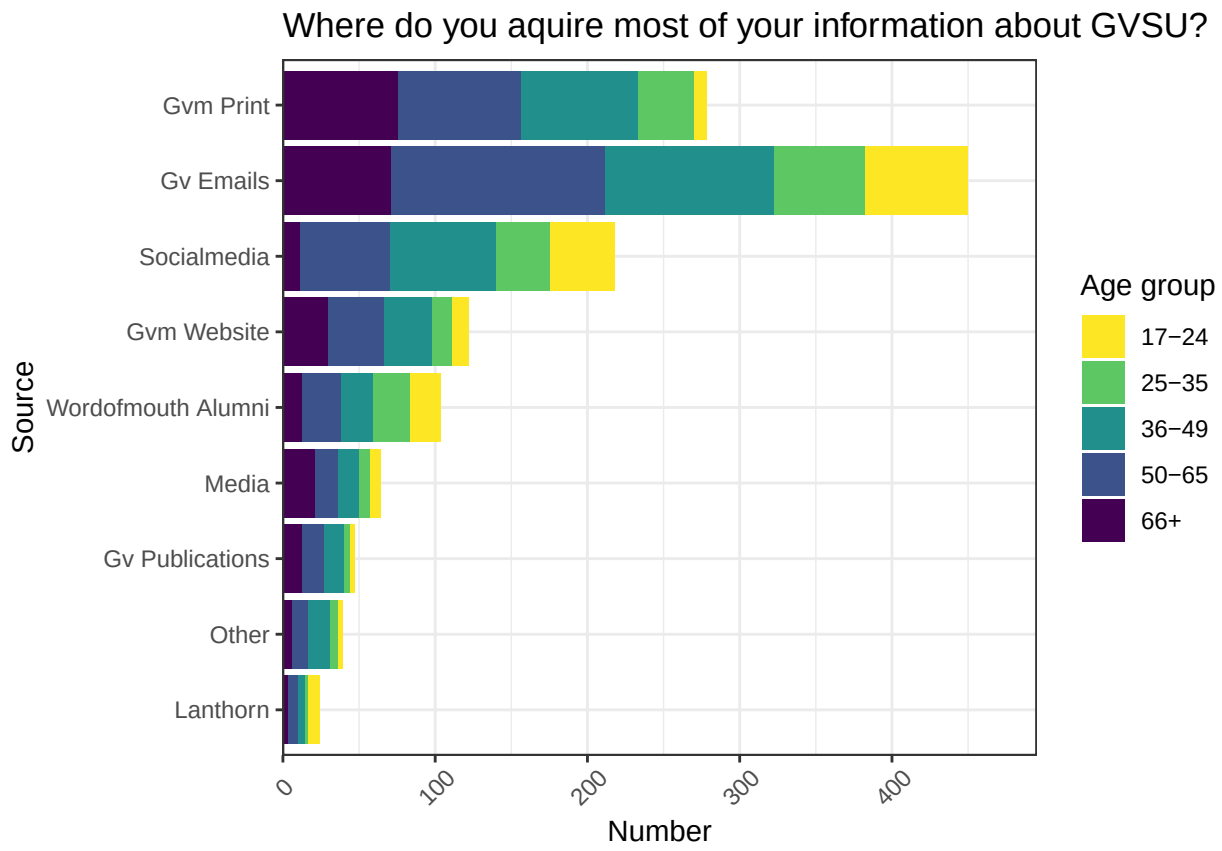
```
## `summarise()` has grouped output by 'key_pop'. You can override using the
## `.groups` argument.
```

```
actions_key_pop |>
  ggplot(aes(key_pop, percent, fill = as.factor(num_actions))) +
  geom_col(position = "dodge") +
  theme_bw() +
```

```
  scale_y_continuous(expand = expansion(mult = c(0, 0.2))) +
  scale_fill_viridis_d() +
  labs(
    y = "Percent",
    fill = "Number of actions",
    title = "Key Population by number of actions") +
  theme(axis.title.x = element_blank())
```

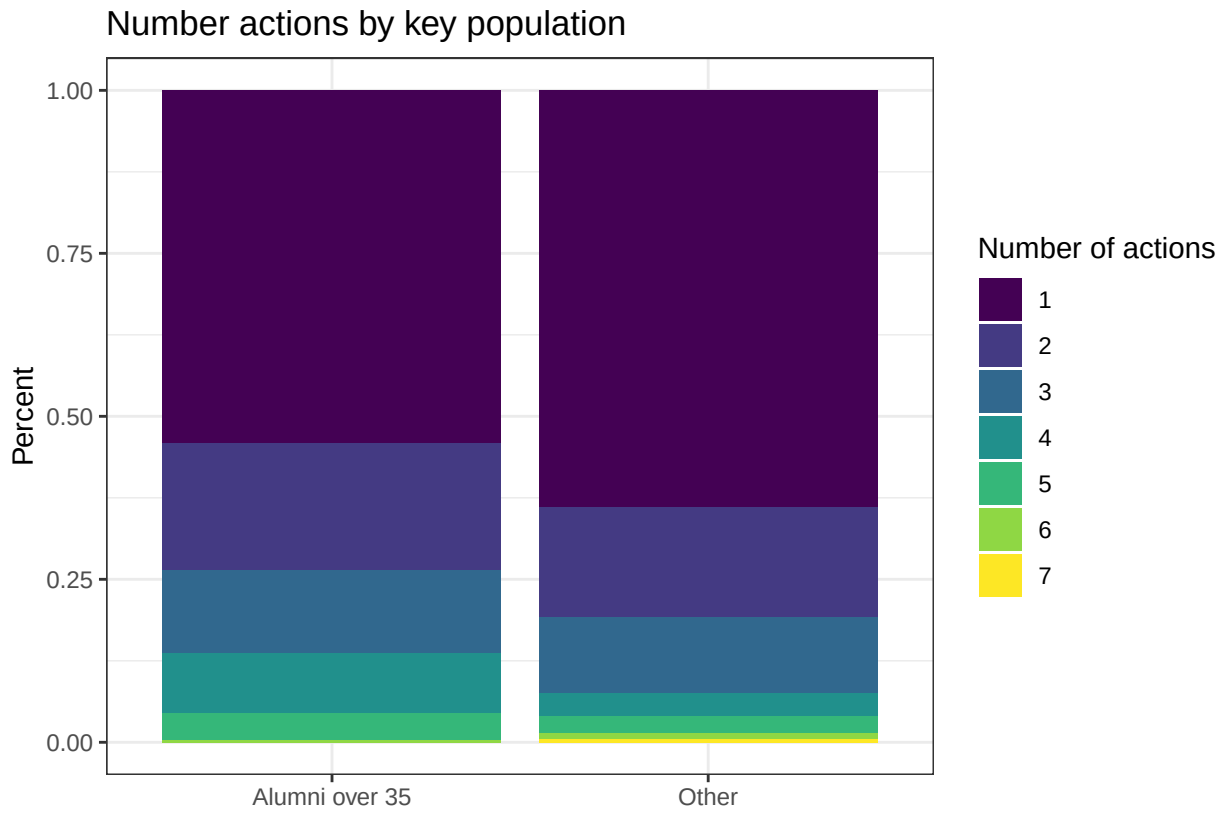



```
age_by_most_info |>
  ggplot(aes(fct_reorder(most_info, n), n)) + # grab variables and reorder action
    geom_col(aes(fill = age))  +
    theme_bw() +
    theme(axis.text.x = element_text(angle = 45, vjust = .6)) +
    labs(
      x = "Source",
      y = "Number",
      fill = "Age group",
      title = "Where do you aquire most of your information about GVSU?"
      ) +
    scale_fill_viridis_d(direction = -1) +
    scale_y_continuous(expand = expansion(mult = c(0, 0.1))) +
    coord_flip()
```



- action by key population

```
gvm_clean |>
  ggplot(aes(key_pop, fill = as.factor(num_actions))) +
  geom_bar(position = "fill") +
  theme_bw() +
  labs(
    x = "",
    y = "Percent",
    fill = "Number of actions",
    title = "Number actions by key population"
  ) +
  scale_fill_viridis_d()
```



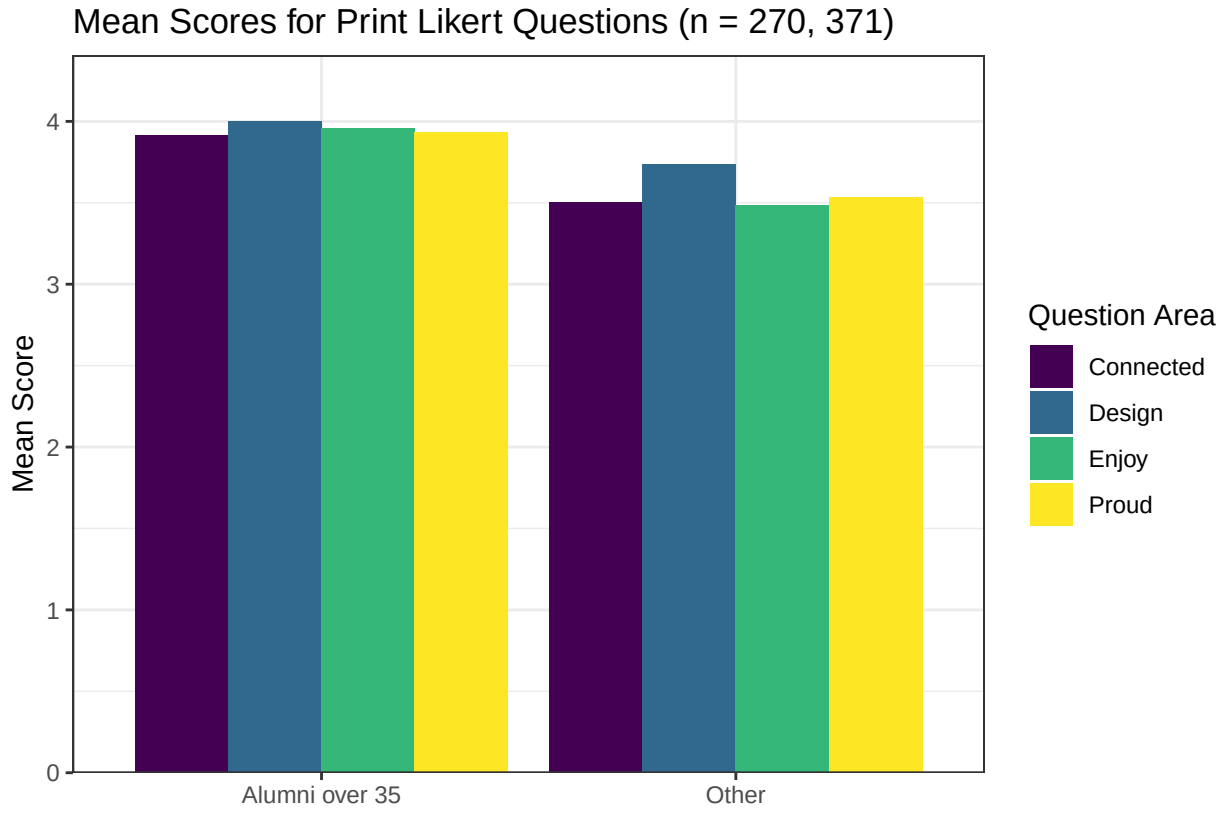
- print likert questions

```
key_print_percents <- gvm_clean |>
  select(key_pop, age, starts_with("print_")) |>
  pivot_longer(!c(key_pop, age), names_to = "likert_type", values_to = "value") |>
  mutate(likert_type = str_replace_all(likert_type, "print_|likert", ""),
         likert_type = str_to_title(str_replace_all(likert_type, "_", " "))) |>
  group_by(key_pop, likert_type) |>
  summarize(count = mean(value),
            n = n())
```

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

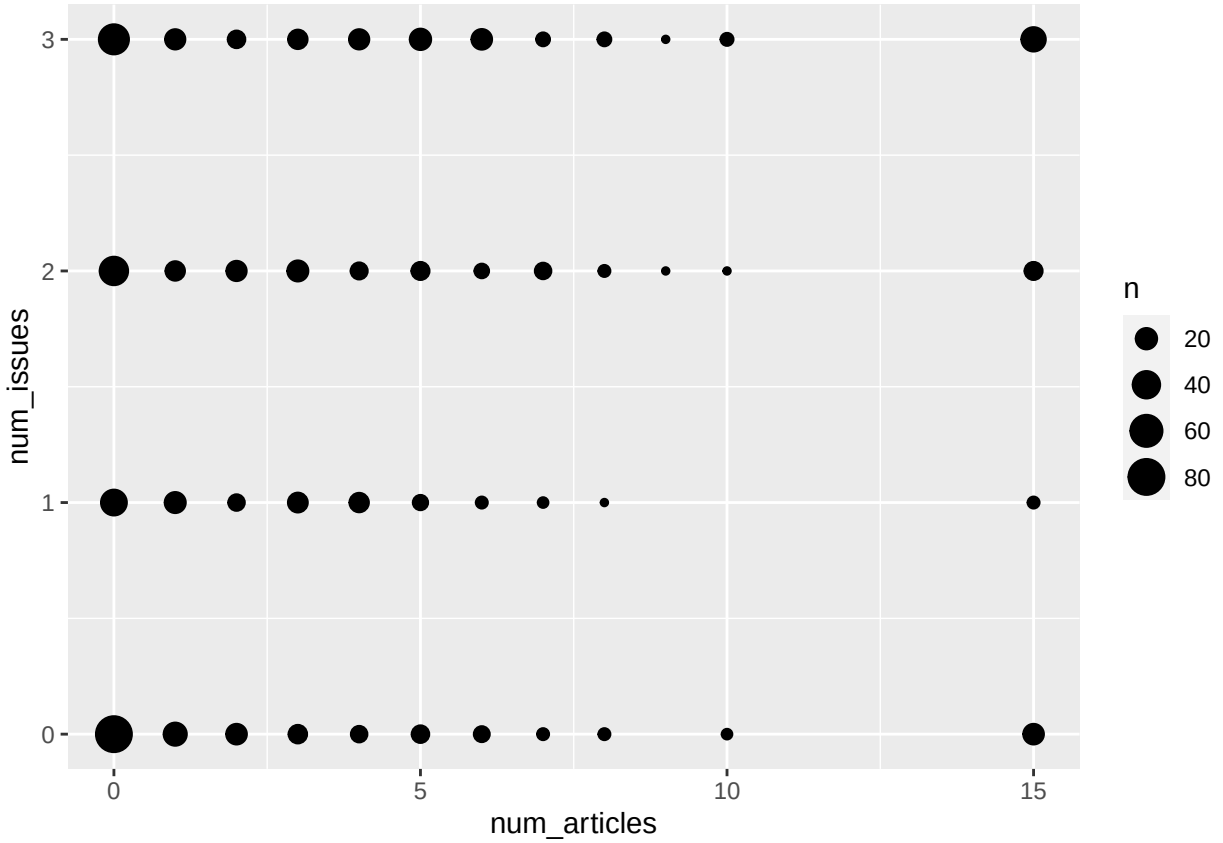
```
key_print_percents |>
  ggplot(aes(key_pop, count, fill = likert_type)) +
  geom_col(position = "dodge") +
  theme_bw() +
  labs(
    x = "",
    y = "Mean Score",
    fill = "Question Area",
    title = paste(
      "Mean Scores for Print Likert Questions ",
      "(n = ", key_print_percents$n[1], ", ", key_print_percents$n[5], ")" ,
      sep = "")) +
  scale_y_continuous(expand = expansion(mult = c(0, 0.1))) +
```

```
  scale_fill_viridis_d()
```

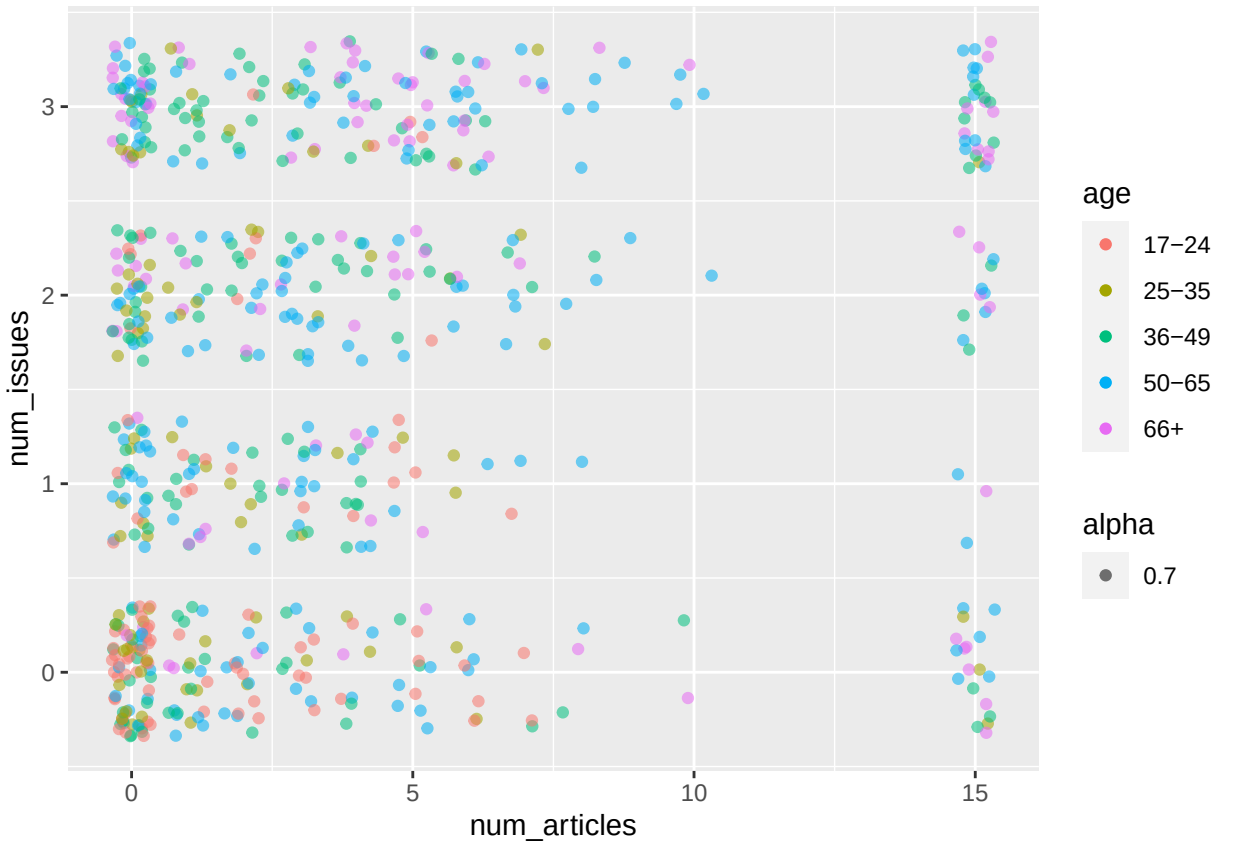


- num_issues and num_articles

```
ggplot(gvm_clean, aes(x = num_articles, y = num_issues))+
  geom_count()
```




```
ggplot(gvm_clean, aes(x = num_articles, y = num_issues))+
  geom_jitter(width =.35, height = .35, aes(color=age, alpha=.7))
```

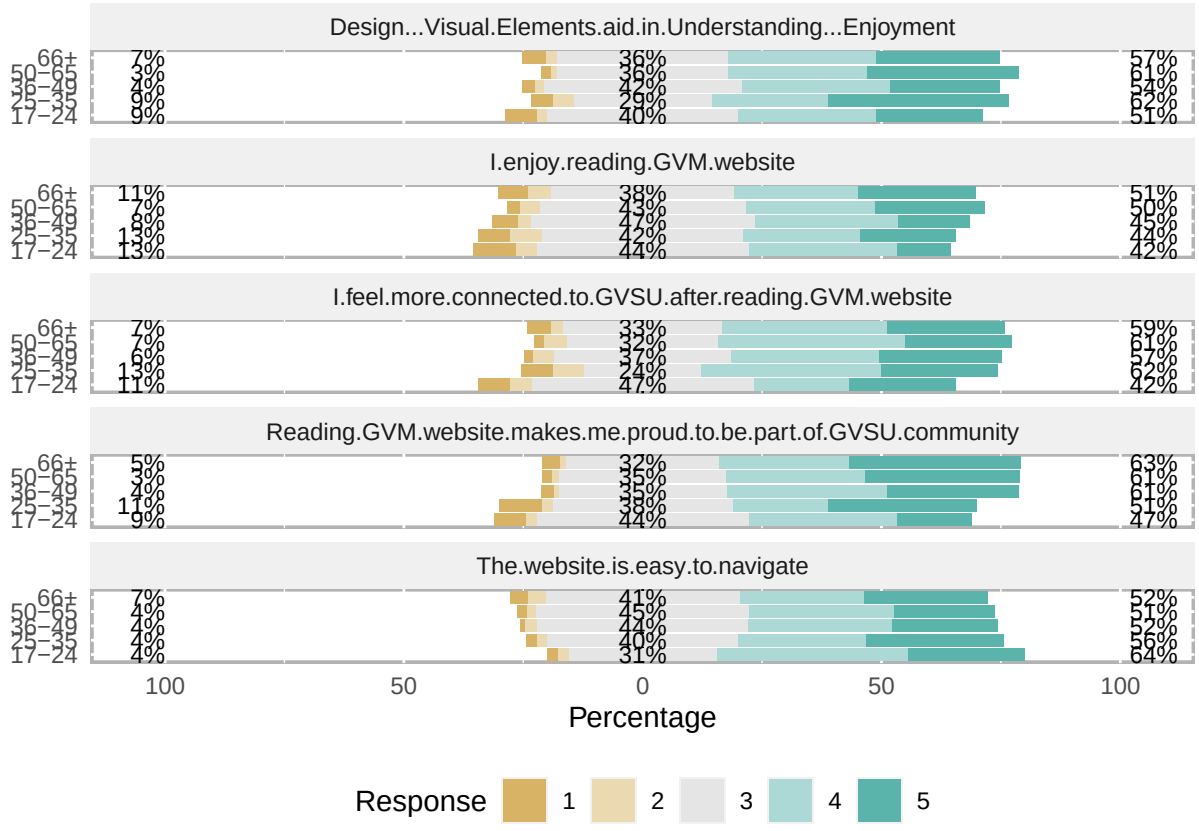



```
# sjPlot::tab_xtab(var.row = gvm_clean$num_articles, var.col = gvm_clean$num_issues, title = "Articles
```

- Website Likert Questions

```
#filtering out responses of those who say they read the website
select_levels <- c("GVM Website","Both")
website_likert_df <- gvm_clean %>%
  filter(where_read %in% select_levels)
rm(select_levels)

#filtering out responses of those who've read more than zero website articles
website_likert_df <- gvm_clean %>%
  filter(num_articles != 0)

#making df for website likert questions
website_likert_df <- select(website_likert_df, starts_with("website_") & ends_with("_likert"), age)

#defining levels
custom_levels <- c("Strongly Disagree", "Slighly Disagree", "Neutral", "Slighly Agree", "Strongly Agree

#renaming relevant columns
website_likert_df <- website_likert_df %>% rename(
  "I enjoy reading GVM website" = website_enjoy_likert,
  "The website is easy to navigate" = website_navigate_likert,
  "Design & Visual Elements aid in Understanding & Enjoyment" = website_design_likert,
```

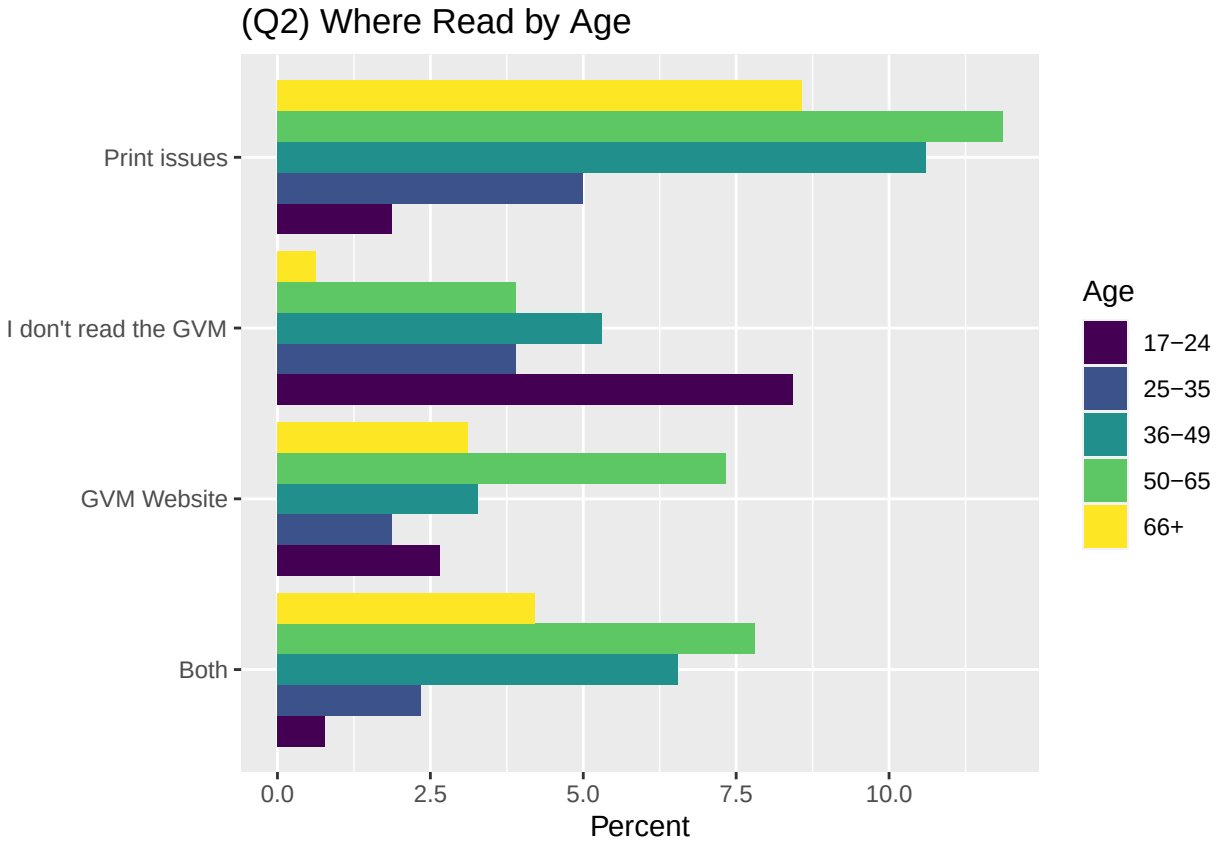
```
  "I feel more connected to GVSU after reading GVM website" = website_connected_likert,
  "Reading GVM website makes me proud to be part of GVSU community" = website_proud_likert)

#making all variables factors
website_likert_df <- data.frame(lapply(website_likert_df, as.factor))

#plotting likert
wl_format <- likert(website_likert_df[,1:5], grouping=website_likert_df[,6])

wl_format %>% plot()
```



Where_Read by Age

```
#Create data frame with the counts of each typ
where_read_by_age <- as.data.frame(table(gvm_clean$age,gvm_clean$where_read))

#renaming
where_read_by_age <- rename(where_read_by_age,'age'='Var1',
                                    'where_read'='Var2',
                                    'Count'='Freq') %>%
                     mutate(percent = (Count/sum(Count))*100)

#Plotting
where_read_by_age %>% ggplot(aes(x=where_read, y=percent, fill=age)) +
```

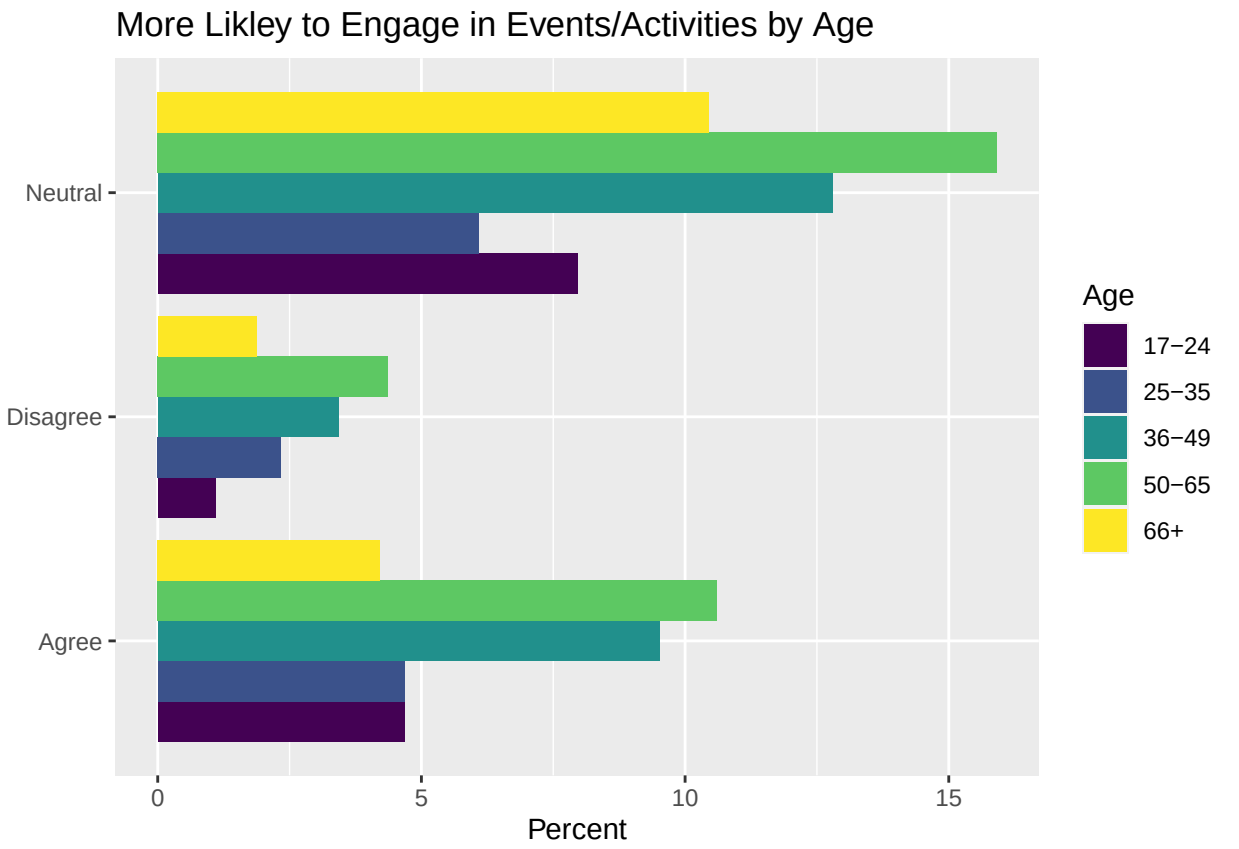
```
  geom_bar(stat = 'identity', position='dodge') +
  labs(title = "(Q2) Where Read by Age",
       y = "Percent",
       fill = "Age") +
  theme(axis.title.y = element_blank()) +
  coord_flip() +
  scale_fill_viridis_d()
```



Engagement bt Age

```
#Create data frame with the counts of each typ
engagement_by_age <- as.data.frame(table(gvm_clean$age,gvm_clean$gvsu_engagement))

#renaming
engagement_by_age <- rename(engagement_by_age,'age'='Var1',
                                    'gvsu_engagement'='Var2',
                                    'Count'='Freq') %>%
                     mutate(percent = (Count/sum(Count))*100)

#Plotting
engagement_by_age %>% ggplot(aes(x=gvsu_engagement, y=percent, fill=age)) +
  geom_bar(stat = 'identity', position='dodge') +
  labs(title = "More Likley to Engage in Events/Activities by Age",
       y = "Percent",
```

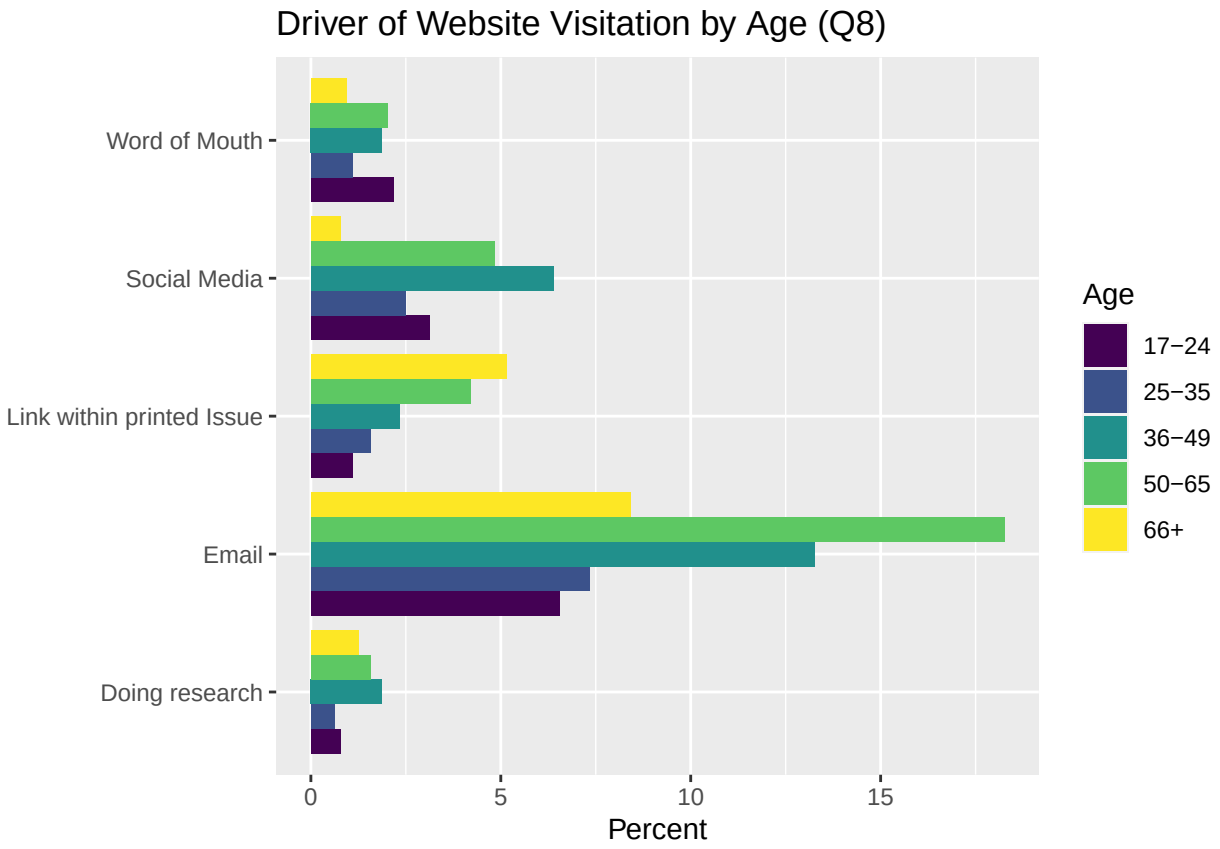
```
      fill = "Age") +
  theme(axis.title.y = element_blank()) +
  coord_flip() +
  scale_fill_viridis_d()
```



drive_to_website by age

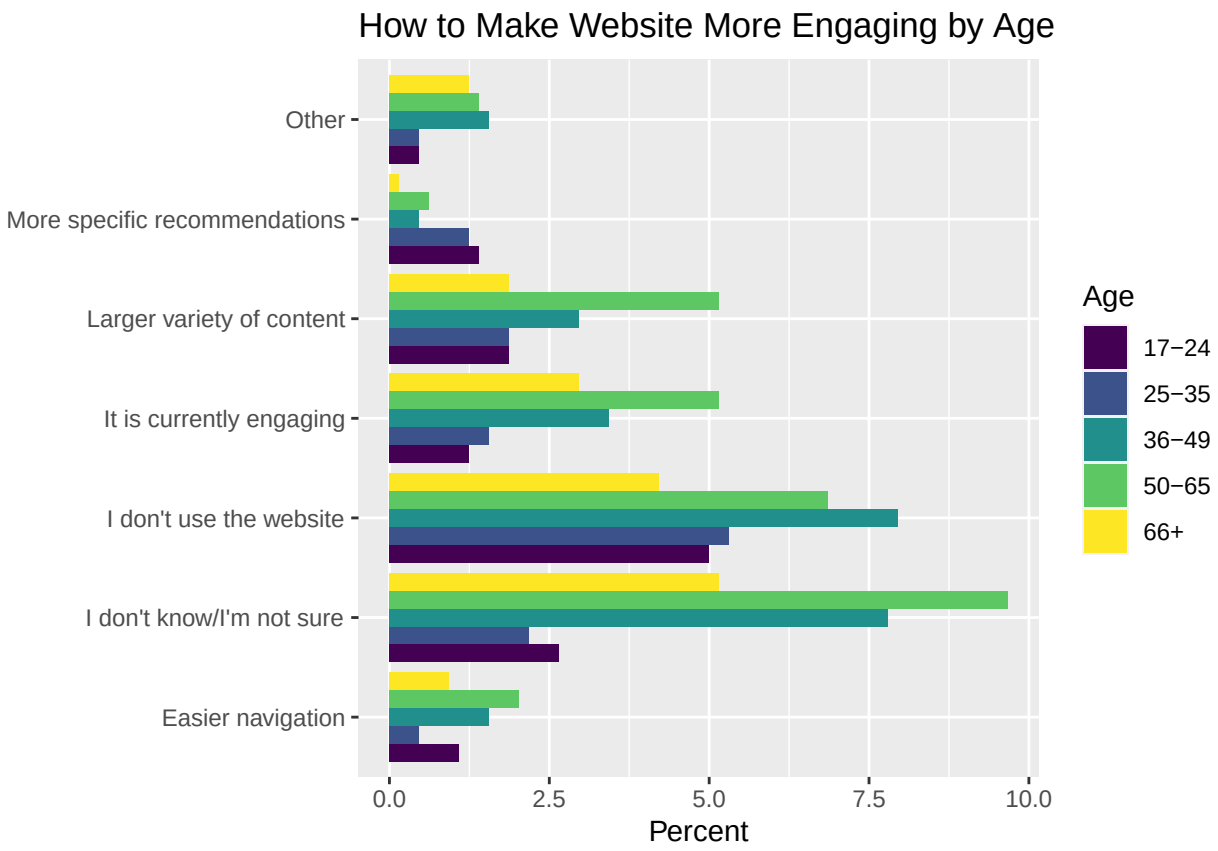
```
#Create data frame with the counts of each typ
drive_to_website_by_age <- as.data.frame(table(gvm_clean$age,gvm_clean$drive_to_website))

#renaming
drive_to_website_by_age <- rename(drive_to_website_by_age,'age'='Var1',
                                  'drive_to_website'='Var2',
                                  'Count'='Freq') %>%
                    mutate(percent = (Count/sum(Count))*100)

#Plotting
drive_to_website_by_age %>% ggplot(aes(x=drive_to_website, y=percent, fill=age)) +
  geom_bar(stat = 'identity', position='dodge') +
  labs(title = "Driver of Website Visitation by Age (Q8)",
       y = "Percent",
       fill = "Age") +
  theme(axis.title.y = element_blank()) +
  coord_flip() +
```

```
  scale_fill_viridis_d()
```



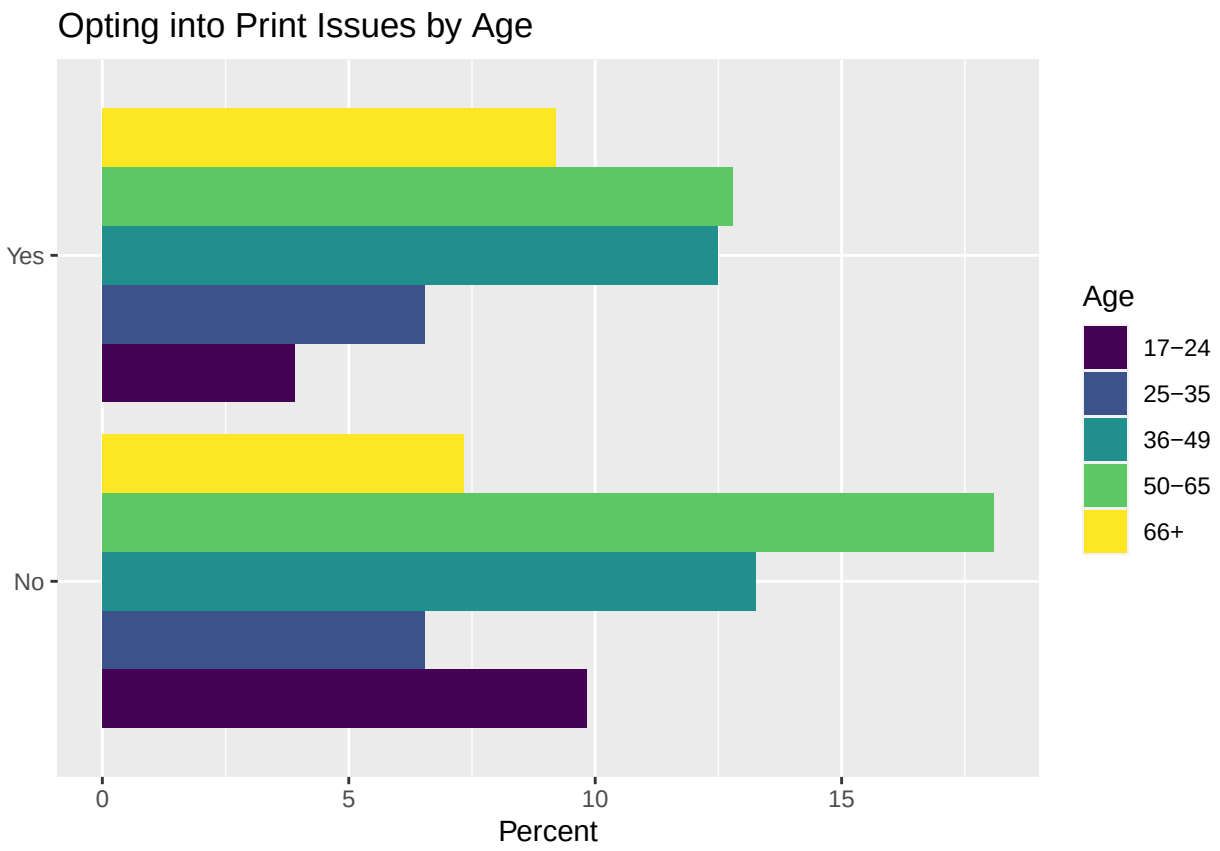
website_engagement by age

```
#Create data frame with the counts of each typ
website_engagement_by_age <- as.data.frame(table(gvm_clean$age,gvm_clean$website_engagment))
#renaming
website_engagement_by_age <- rename(website_engagement_by_age,'age'='Var1',
                                    'website_engagement'='Var2',
                                    'Count'='Freq') %>%
                     mutate(percent = (Count/sum(Count))*100)

#Plotting
website_engagement_by_age %>% ggplot(aes(x=website_engagement, y=percent, fill=age)) +
  geom_bar(stat = 'identity', position='dodge') +
  labs(title = "How to Make Website More Engaging by Age",
       y = "Percent",
       fill = "Age") +
  theme(axis.title.y = element_blank()) +
  coord_flip() +
  scale_fill_viridis_d()
```



Opting by age

```
#Create data frame with the counts of each typ
opting_by_age <- as.data.frame(table(gvm_clean$age,gvm_clean$opting))

#renaming
opting_by_age <- rename(opting_by_age,'age'='Var1',
                                      'opting'='Var2',
                                      'Count'='Freq') %>%
                    mutate(percent = (Count/sum(Count))*100)

#Plotting
opting_by_age %>% ggplot(aes(x=opting, y=percent, fill=age)) +
  geom_bar(stat = 'identity', position='dodge') +
  labs(title = "Opting into Print Issues by Age",
       y = "Percent",
       fill = "Age") +
  theme(axis.title.y = element_blank()) +
  coord_flip() +
  scale_fill_viridis_d()
```



Exporting clean data as GVM_data.csv

```
write_csv(gvm_clean, "GVM_data.csv")
```