Lab Module 3: Bar Charts

PHW251B: Data Visualization for Public Health

Contents

Introduction 2

The Data 3

Data Pre-processing 4

Part 1: Bar Chart Basic Anatomy **5**

1.1: Setting up the initial `ggplots` layer 5

1.2: Titles, captions, axes labels, fonts 6

Part 2: Color **8**

2.1: Adding color 8

2.2: Color Use Cases 14

2.3: Legend 17

2.4: Gridlines/background 19

Part 3: Types of bar charts **26**

3.1: Stacked bar chart 27

3.2: Dodged bar charts 39

3.3: Faceted bar charts 44

3.4: Horizontal bar chart 50

Assignment: Module Questions **53**

Question 1: Create Bar Chart 53

Question 2: Create Stacked Bar Chart 54

Question 3: Create Dodged Bar Chart 55

Introduction

Welcome to Week 3!

In this module, we will explore how to utilize `R` and the `ggplot2` package to create publication-ready bar charts. Bar charts are useful in visualizing summary of data by categories.

After this module, students should be able to:

- Create basic graphs using `ggplot`
- Use advanced `ggplot` aesthetics to customize graphs
- Apply concepts of effective graphical design to produce professional graphs

Helpful Resources:

- ggplot Cheatsheet link
- R Brewer Palettes link
- HEX Color Palette Generator link
- ggthemes List link

The Data

For this module, we will use data from a case-control study of esophageal cancer in Ille-et-Vilaine, France using a built-in dataset in R.

There are 5 variables:

- **agegp**: Age group
- **alcgp**: Alcohol consumption in grams/day
- **tobgp**: Tobacco consumption in grams/day
- **ncases**: Number of cases
- **ncontrols**: Number of controls

```
# Load data
# NOTE: How is loading this built-in data different from loading other datasets?
data(esoph)

# Let's view the first couple of lines of the data
# NOTE: What do you notice about the types of variables?
head(esoph)
```

```
##   agegp     alcgp    tobgp ncases ncontrols
## 1 25-34 0-39g/day 0-9g/day      0        40
## 2 25-34 0-39g/day    10-19      0        10
## 3 25-34 0-39g/day    20-29      0         6
## 4 25-34 0-39g/day      30+      0         5
## 5 25-34     40-79 0-9g/day      0        27
## 6 25-34     40-79    10-19      0         7
```

Data Pre-processing

Based on the data, I am curious to see which age group has the most cases of esophageal cancer. So, to find that out, we need to first clean and subset our data.

```
# The 'agg_agegp' data frame will contain summarized information about esophageal cancer cases by age g
agg_agegp <- esoph %>%
  # Group the data by the 'agegp' variable
  group_by(agegp) %>%
  # Summarize the grouped data by calculating the total number of cases (ncases) for each age group
  summarize(totalcases = sum(ncases))
# Display or return the 'agg_agegp' data frame, which now contains the summarized information
agg_agegp
```

```
## # A tibble: 6 x 2
##   agegp totalcases
##   <ord>      <dbl>
## 1 25-34          1
## 2 35-44          9
## 3 45-54         46
## 4 55-64         76
## 5 65-74         55
## 6 75+           13
```

We have the total number of cases by age group, but it may be helpful to also get a percentage of the total cases between age groups.

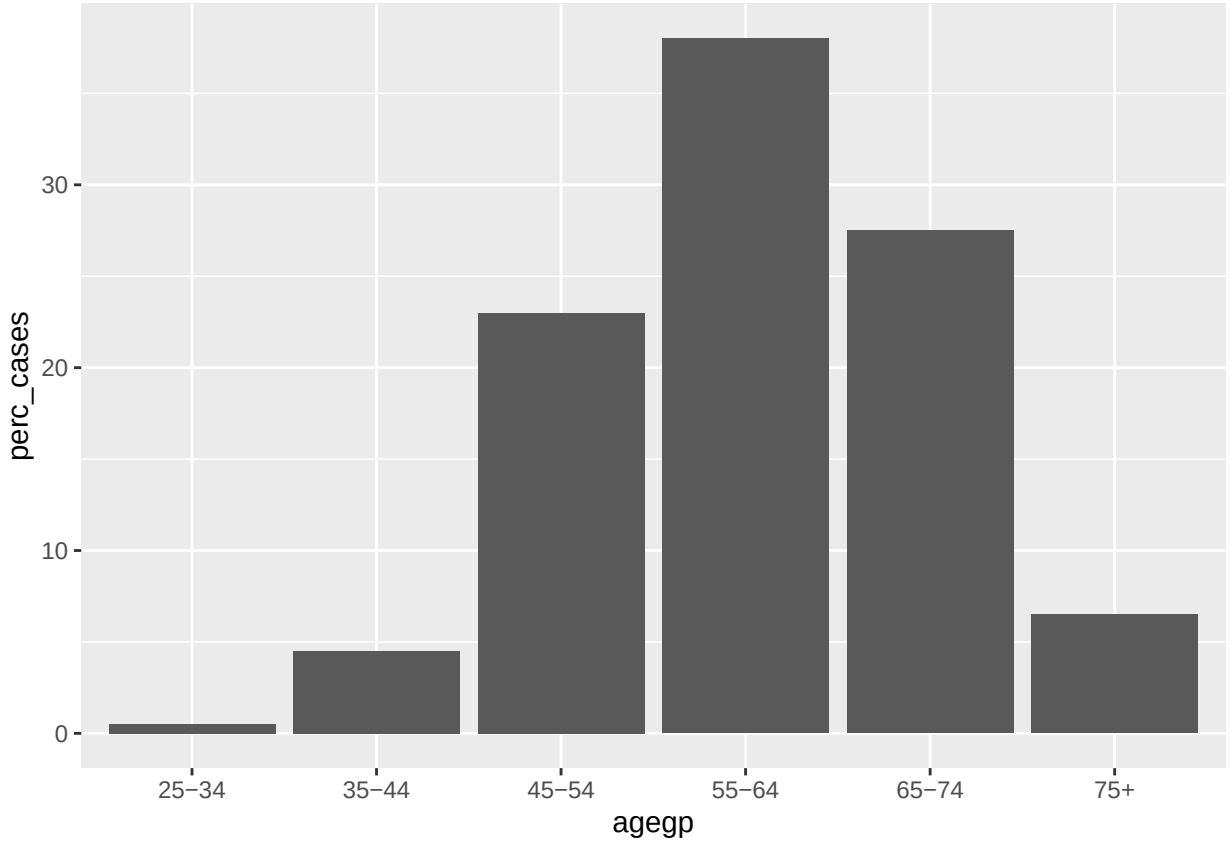
```
# Add a new column 'perc_cases' to the data frame
  # Calculate the percentage of total cases for each age group
  # The calculation is done by dividing 'totalcases' by the sum of all 'totalcases' values, and then mu
agg_agegp <- agg_agegp %>%
  mutate(perc_cases = 100*totalcases/sum(totalcases))
```

To get a better understanding of this trend, let's visualize this data using a barchart.

Part 1: Bar Chart Basic Anatomy

1.1: Setting up the initial ggplots layer

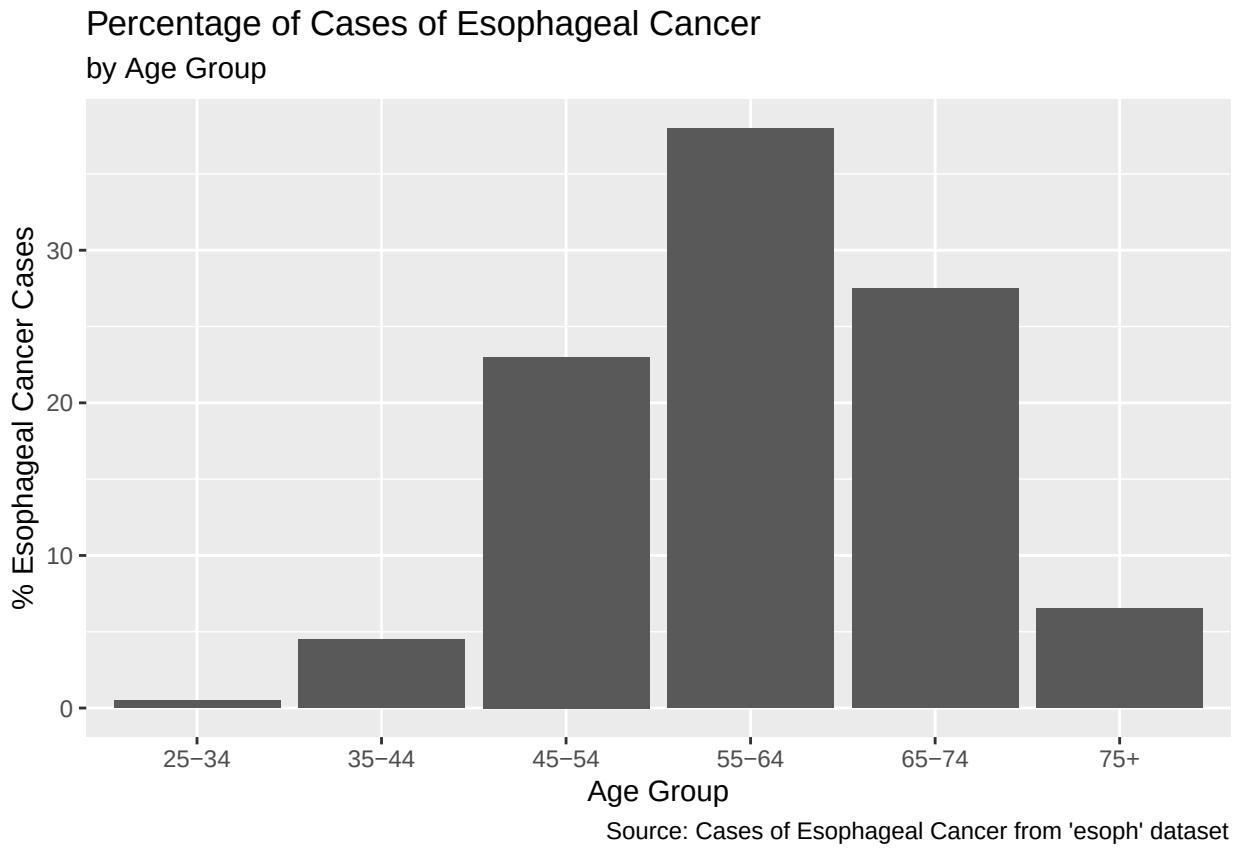
```
# Initialize the 'barchart' object and specify the data frame 'agg_agegp' as the data source
# Use 'aes()' to define the aesthetics (mapping of variables to visual properties)
barchart <- ggplot(data = agg_agegp, # Selecting the data that will be fed into the plot
                   aes(x = agegp, # 'x=agegp' maps the 'agegp' variable to the x-axis
                       y = perc_cases)) + # 'y=perc_cases' maps the 'perc_cases' variable to the y-axis

  # Add a bar layer to the plot using 'geom_bar()'
  # 'stat="identity"' means the heights of the bars correspond to the actual data values
  geom_bar(stat = "identity") # 'geom_bar' specifies that this will be a bar chart

# Display the bar chart
barchart
```



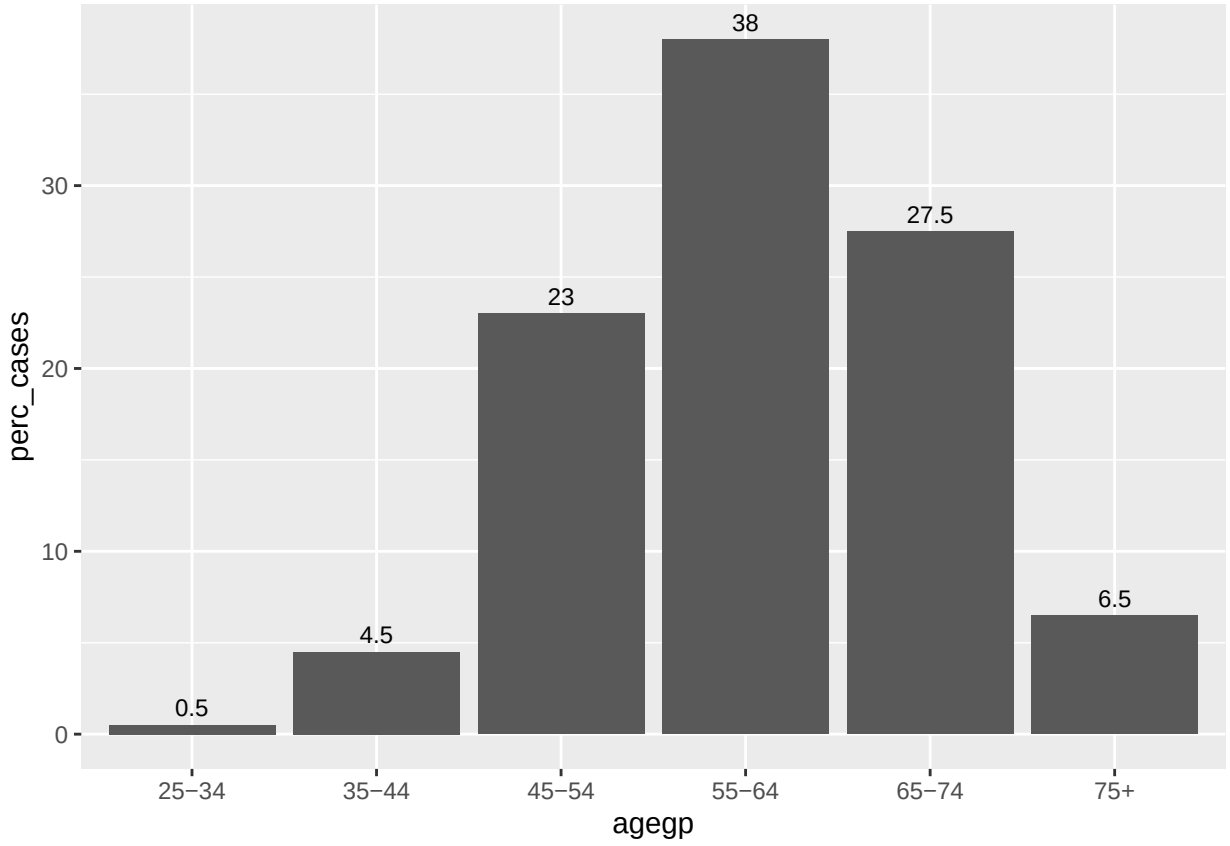
Alright, so we have a basic bar chart! What correlations do you see between the cases of esophageal cancer and age group? We will now move into various aspects of the graph you may want to modify to make it more presentable to an audience.

1.2: Titles, captions, axes labels, fonts

```
# Add custom labels to the 'barchart'
barchart +
  labs(title = "Percentage of Cases of Esophageal Cancer", # Title
       subtitle = "by Age Group", # Subtitle
       x = "Age Group", # X-axis label
       y = "% Esophageal Cancer Cases", # Y-axis label
       caption = "Source: Cases of Esophageal Cancer from 'esoph' dataset") # Caption
```



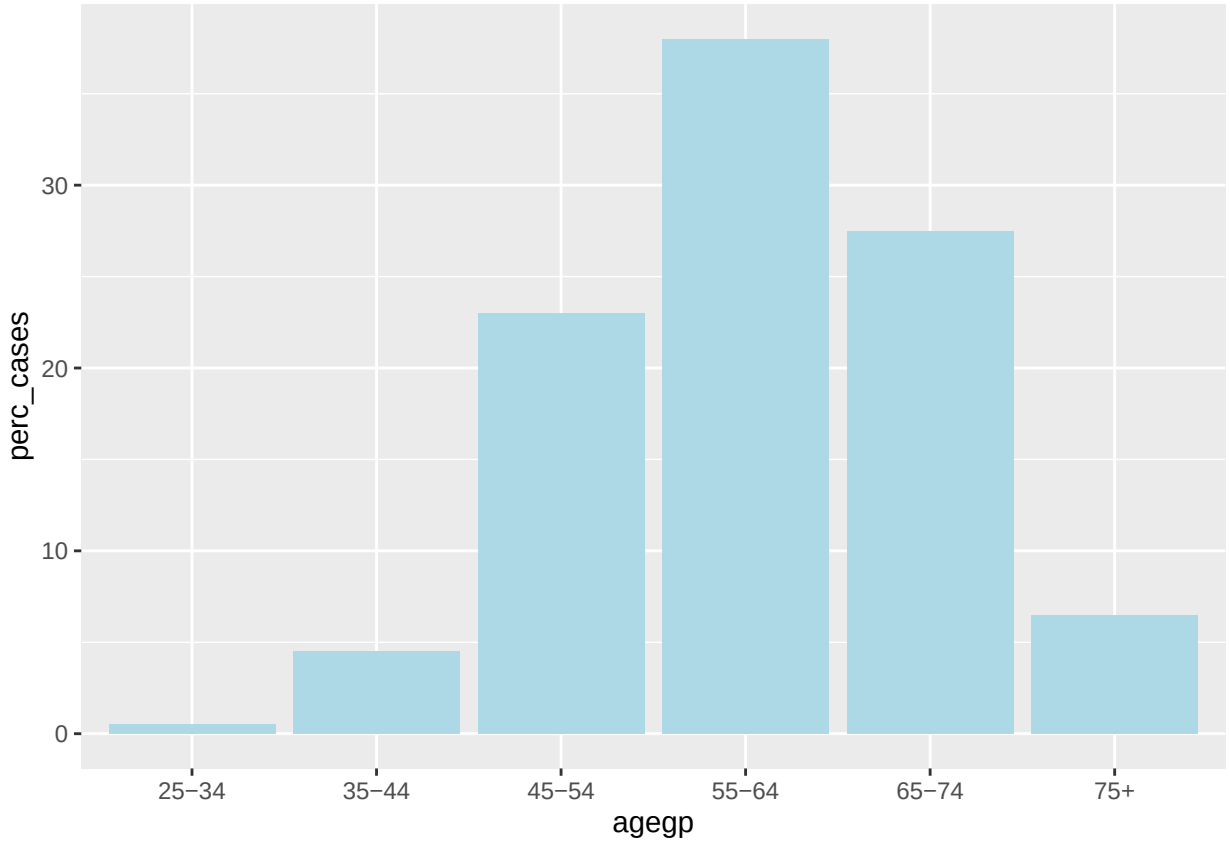

```
# Add text labels above each bar using the 'perc_cases' values as labels
barchart +
  geom_text(aes(label = perc_cases), # Add text labels to each bar
                vjust = -0.5, # Adjust vertical placement (- is up, + is down)
                size = 3, # Adjust text size
                color = "black", # Set color
                family = "sans" # Set font
            )
```


38
27.5
23
6.5
4.5
0.5
30
20
10
0
perc_cases
25−34
35−44
45−54
55−64
65−74
75+
agegp

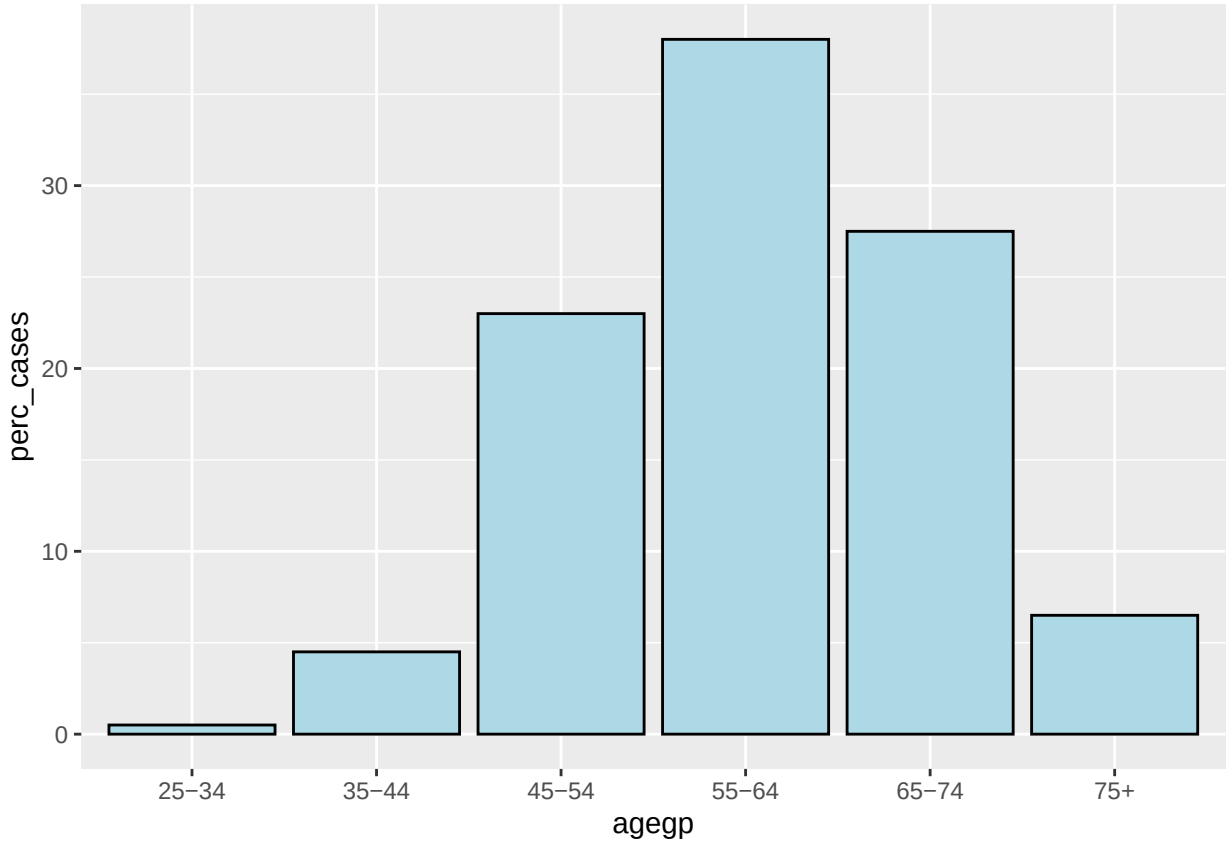
Part 2: Color

2.1: Adding color

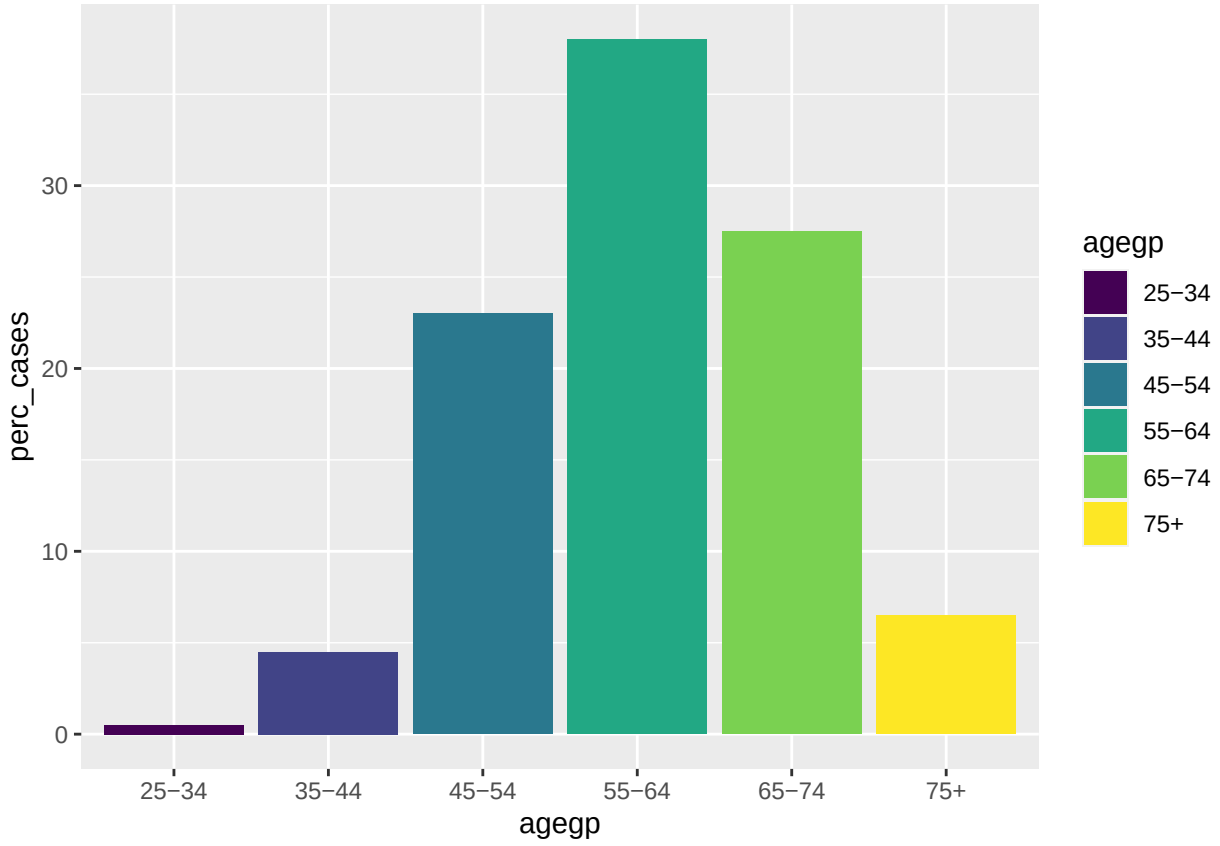
```
# One color for all bars
barchart +
  geom_bar(stat="identity", # Directly plots provided data values as bar heights, without transformatio
           fill="lightblue") # Filling in the bars
```



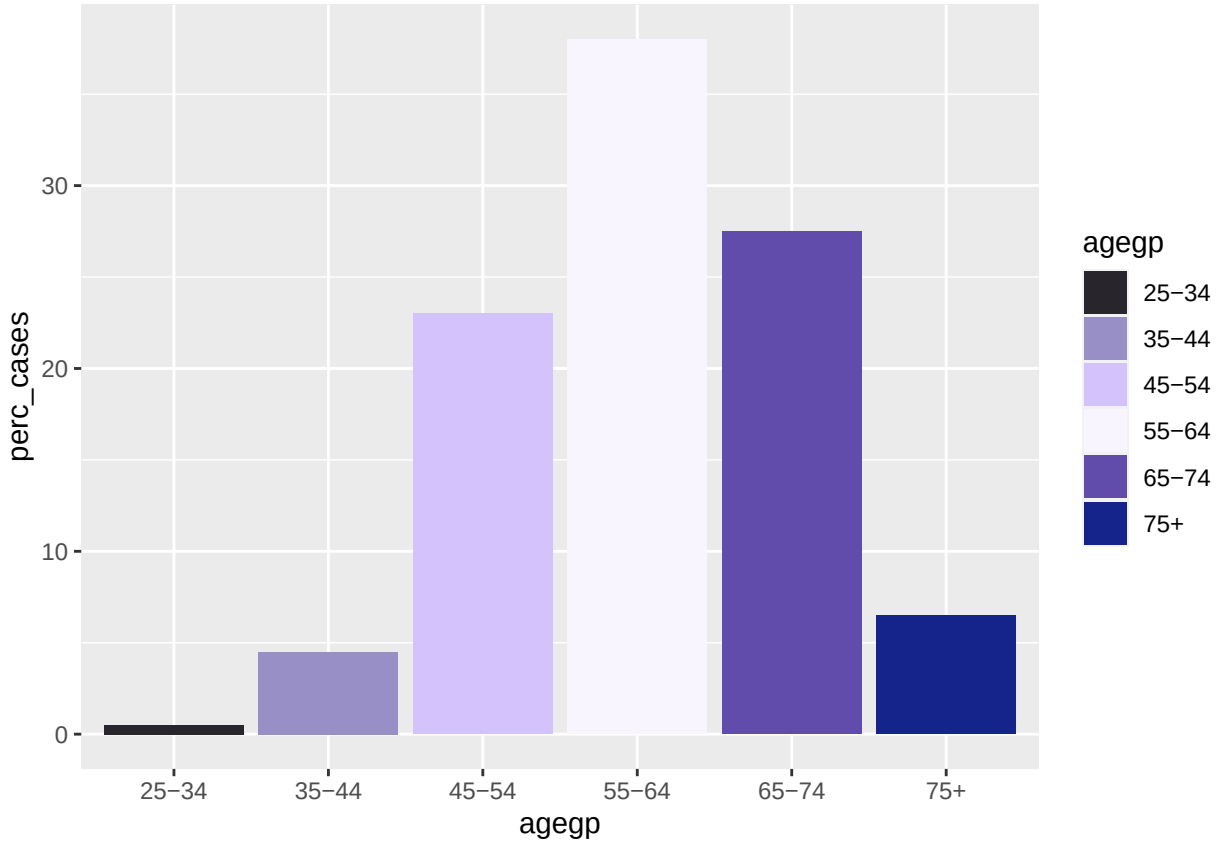

```
# Outline and fill in the bars
barchart +
  geom_bar(stat = "identity",
           fill = "lightblue",
           color = "black") # Outlining the bars
```



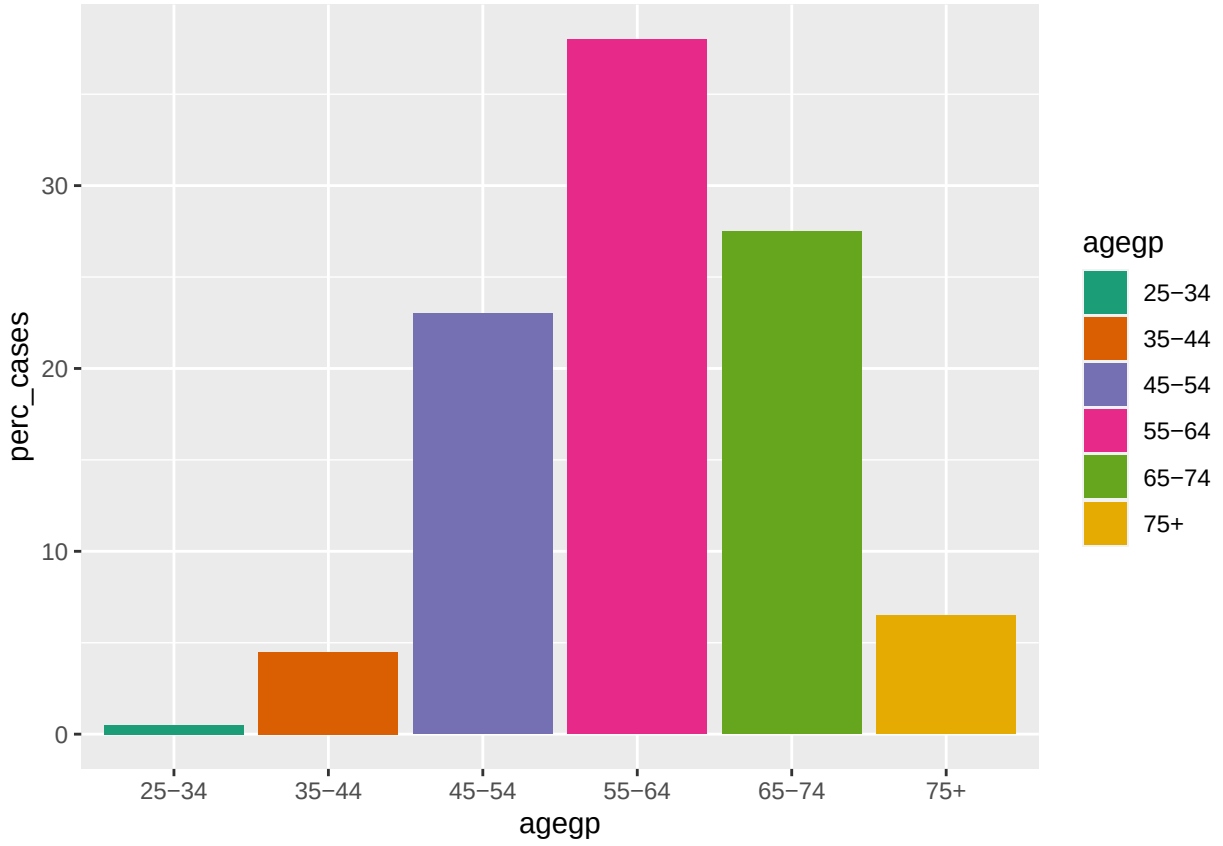

```
# Color by Group (More useful for stacked barcharts which are covered later in this module)

# Use default colors:
barchart2 <- ggplot(agg_agegp, aes(x=agegp, y=perc_cases, fill = agegp)) +
  geom_bar(stat="identity")
barchart2
```



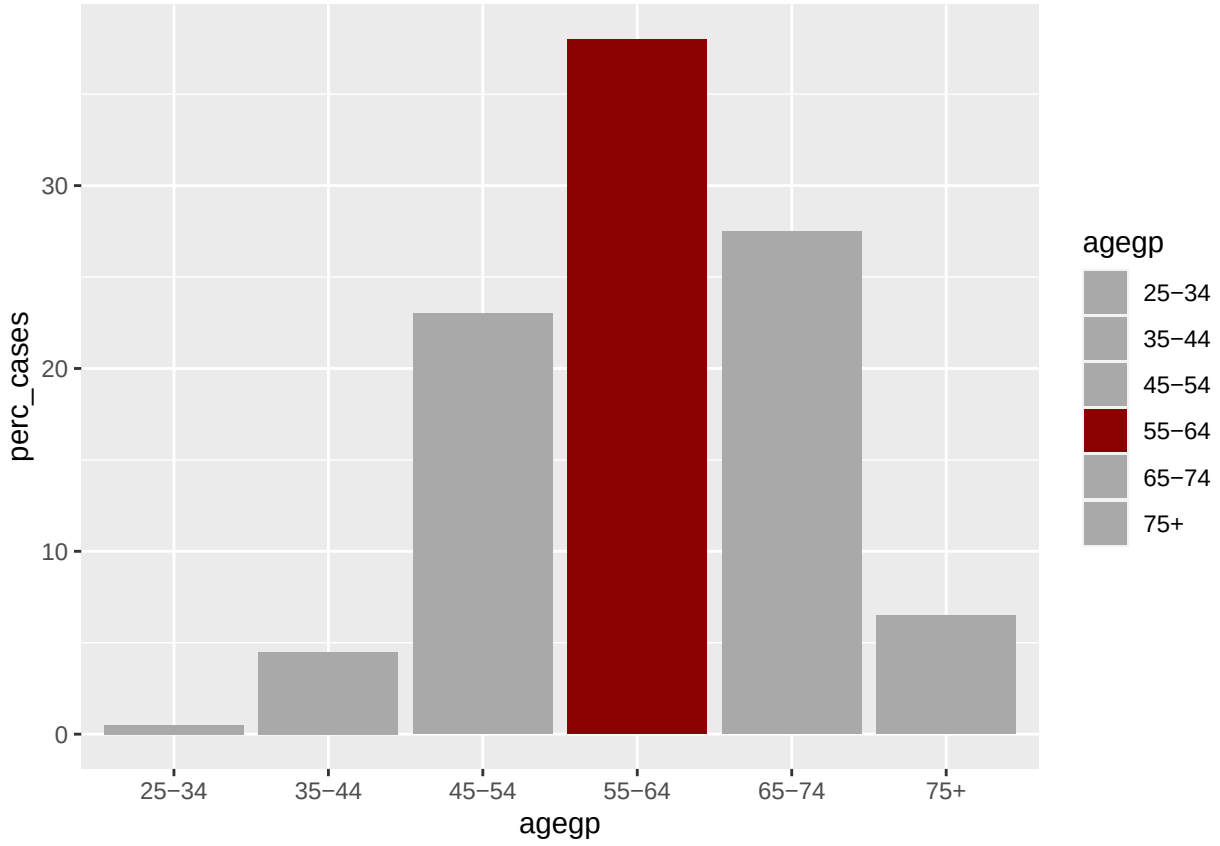

```
# We can also manually select the fill colors:
   # (1) HEX Colors
     barchart2 + scale_fill_manual(values=c("#28262C", "#998FC7", "#D4C2FC",
                                            "#F9F5FF", "#624CAB", "#14248A"))
```




```
# (2) Color Palettes (i.e. Brewer's)
  barchart2 + scale_fill_brewer(palette="Dark2")
```



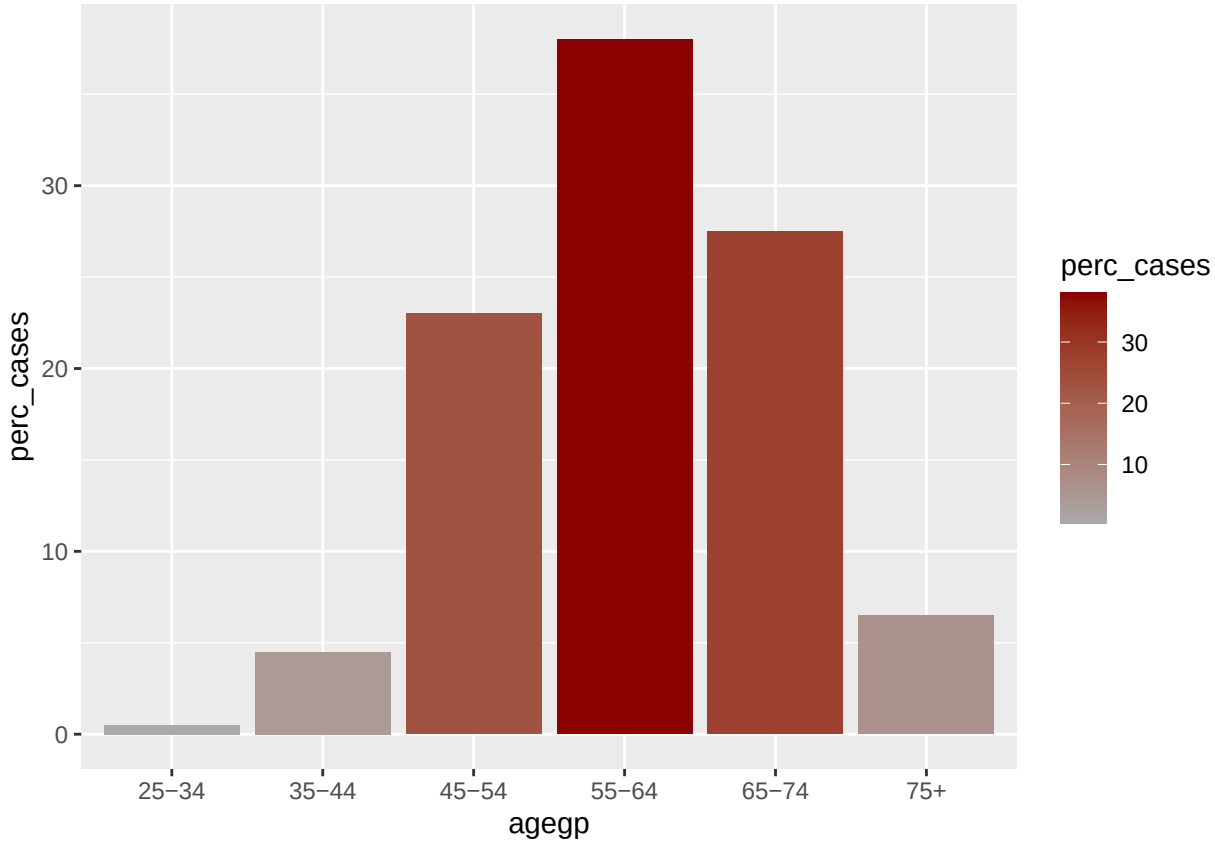

```
# A useful use case is to emphasize one of the groups
    barchart2 + scale_fill_manual(values=c("darkgrey", "darkgrey", "darkgrey",
                                           "darkred", "darkgrey", "darkgrey"))
```


perc_cases
30
20
10
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
agegp
25−34
35−44
45−54
55−64
65−74
75+

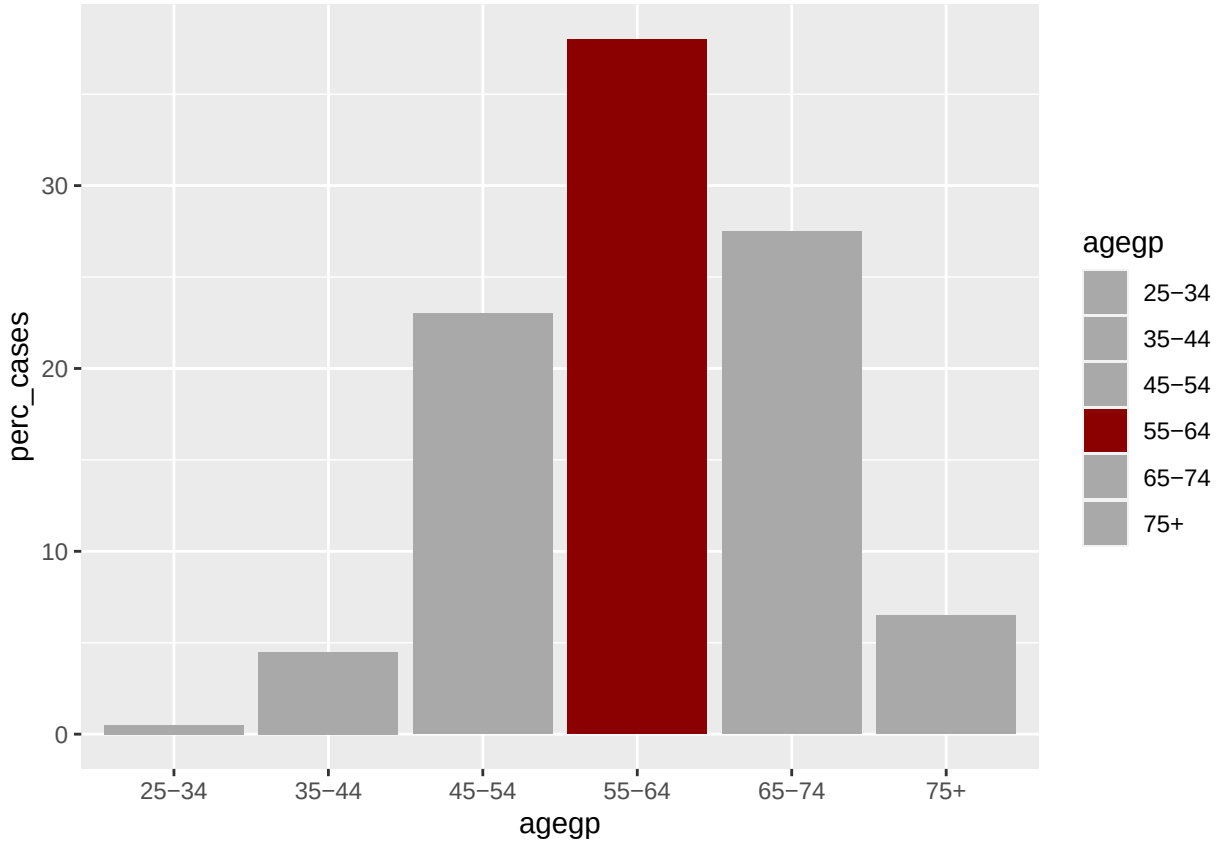
2.2: Color Use Cases

2.2.1: Color to depict quantity (When you want the emphasis to be on a continuous value)

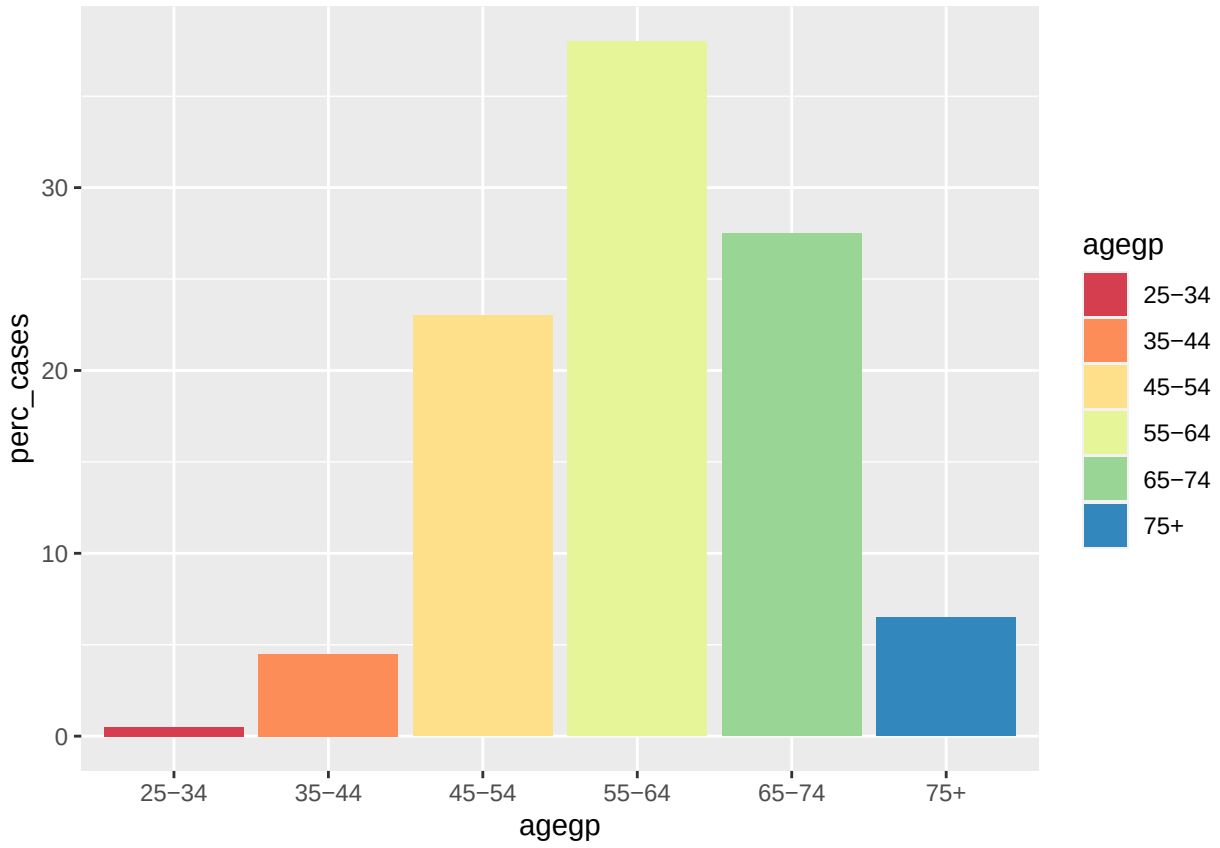
```
barchart_continuous <- ggplot(agg_agegp, aes(x = agegp,
                                  y = perc_cases)) + # Key is to fill with continuous value
  geom_bar(stat = "identity", aes(fill = perc_cases)) +
  scale_fill_gradient(low = "darkgrey", high = "darkred")  # Apply color gradient
barchart_continuous
```




```
# Color to highlight a specific group (When you want to emphasize one of the groups)
barchart2 +
  scale_fill_manual(values=c("darkgrey", "darkgrey", "darkgrey",
                             "darkred", "darkgrey", "darkgrey"))
```

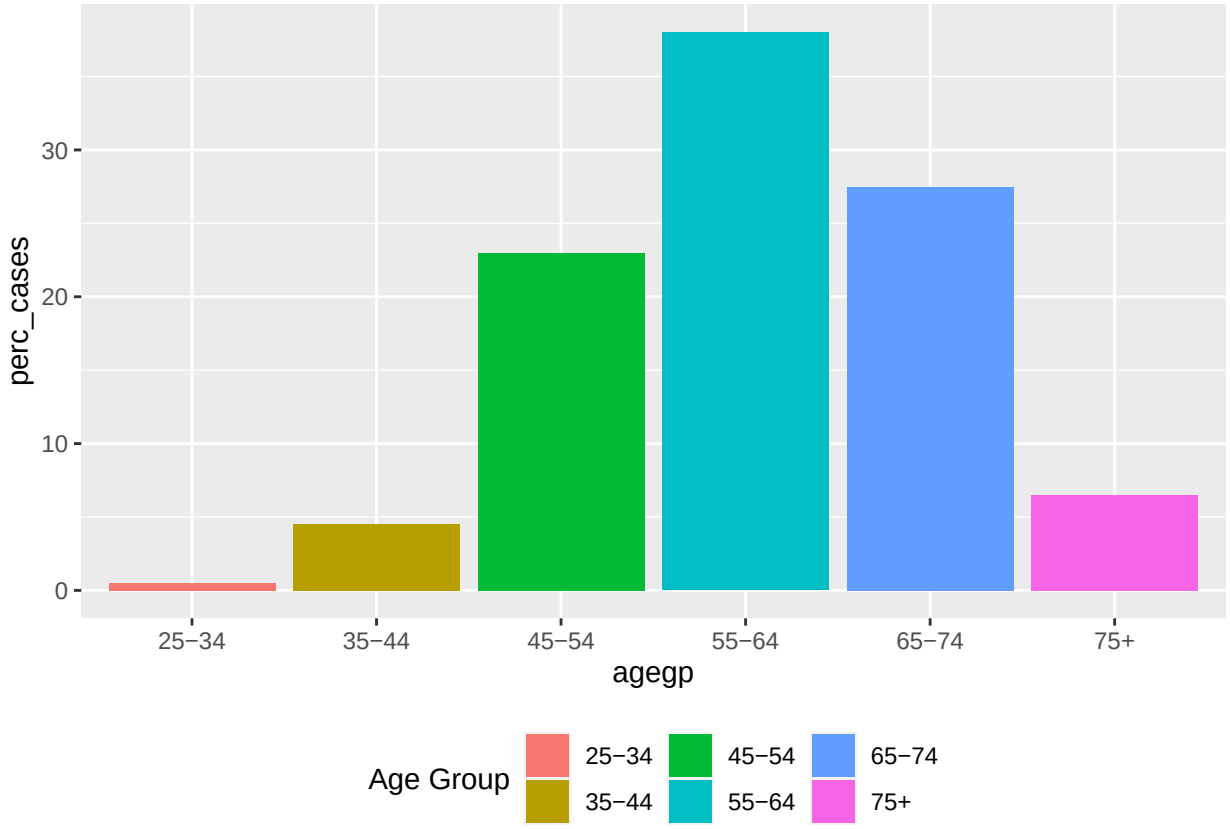



```
# Color to distinguish between groups (Will become more useful in Stacked & Dodged bar plots!!)
barchart2 +
  scale_fill_brewer(palette="Spectral") # Use 'qualitative' or 'diverging' palettes for categorical dat
```

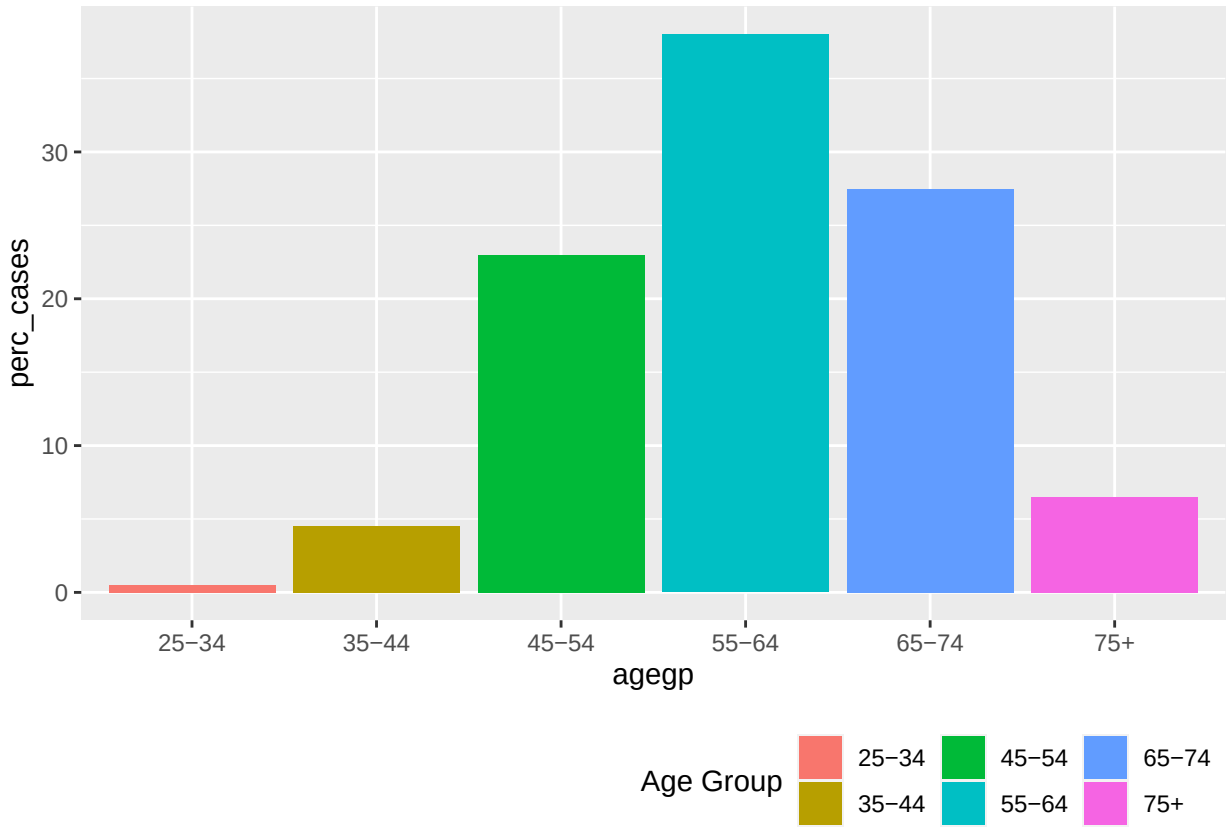

perc_cases
30
20
10
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
agegp
25−34
35−44
45−54
55−64
65−74
75+

2.3: Legend

```
# To modify the Legend:
barchart2 +
  scale_fill_brewer(palette="Accent") +
  scale_fill_discrete(name = "Age Group") + # Change name of Legend
  theme(legend.position="bottom") # Change position of legend (left, right, top, bottom)
```

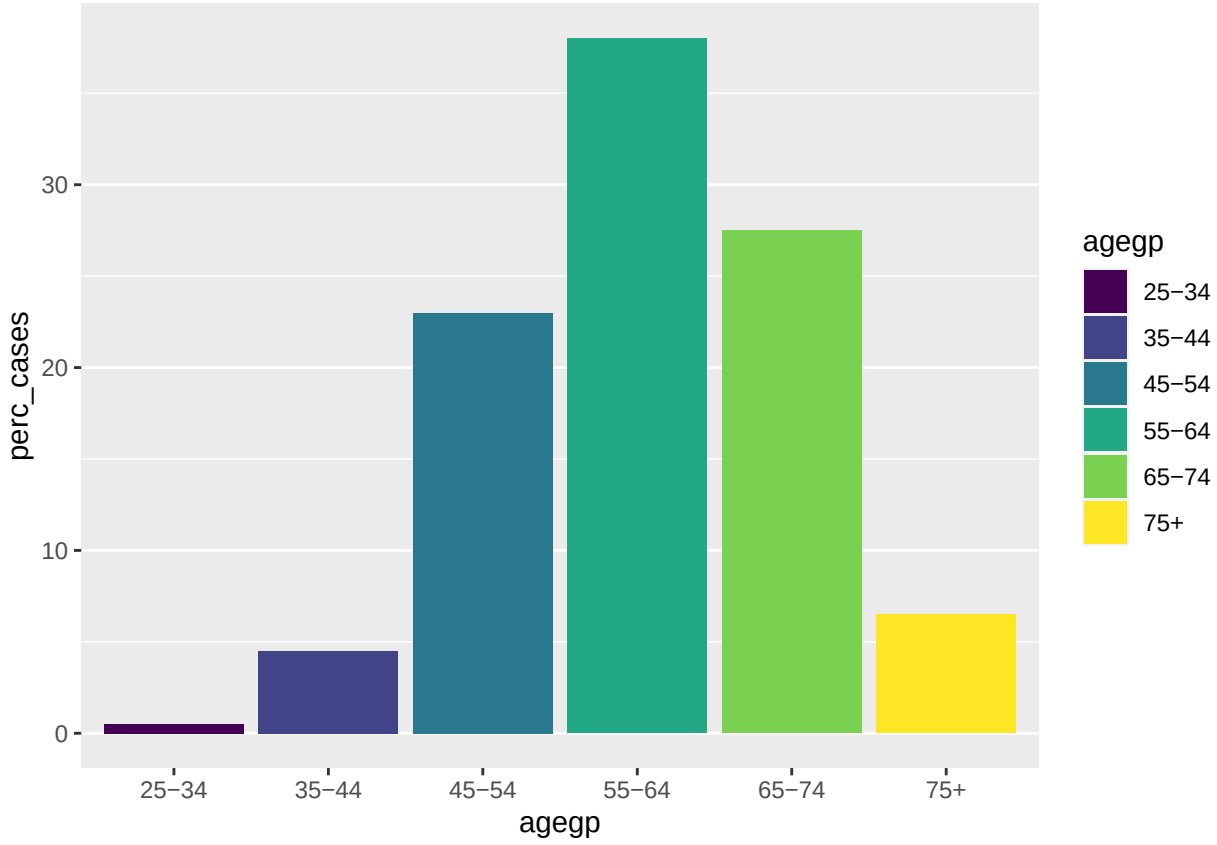



```
barchart2 +
  scale_fill_brewer(palette = "Accent") +
  scale_fill_discrete(name = "Age Group") + # Change name of Legend
  theme(legend.position = "bottom",
        legend.justification = "right") # Another directioning position for combinations (top & right)
```

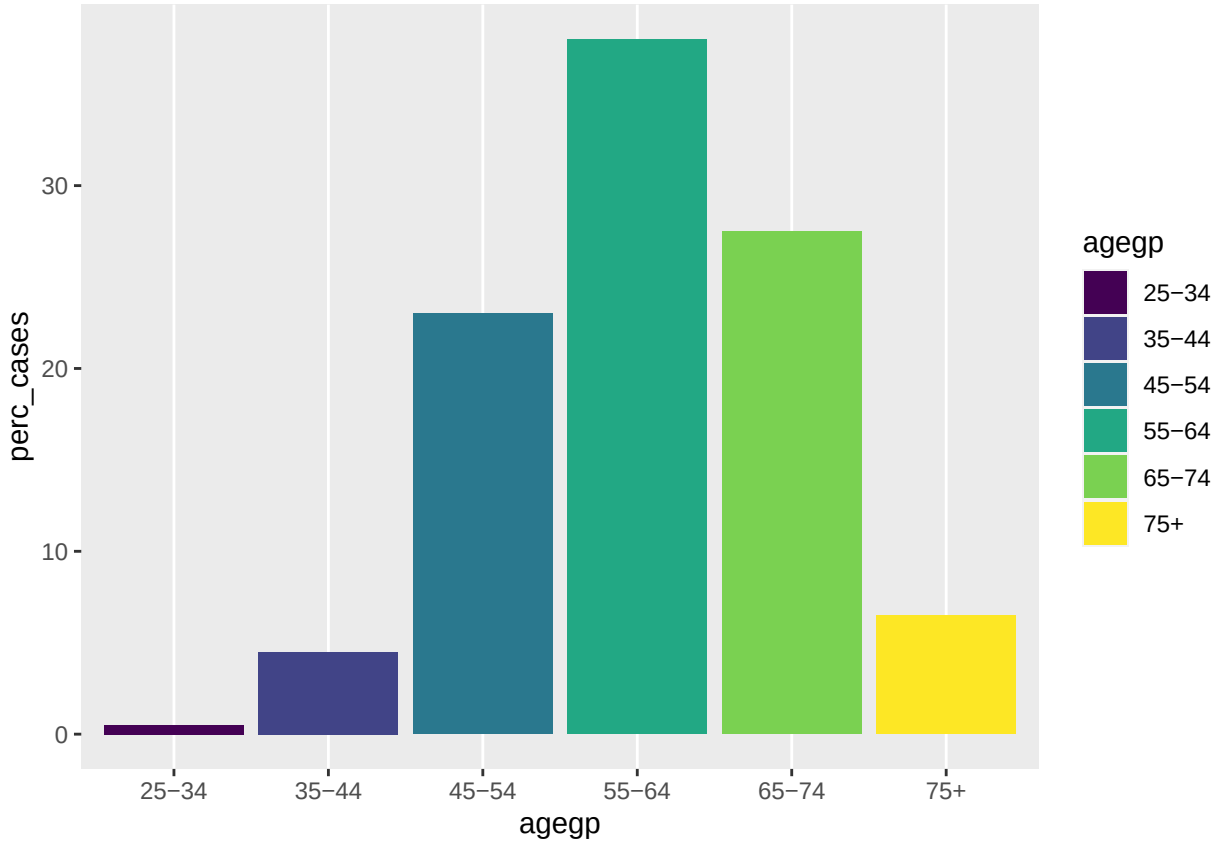

perc_cases
30
20
10
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
Age Group
25−34
35−44
45−54
55−64
65−74
75+

2.4: Gridlines/background

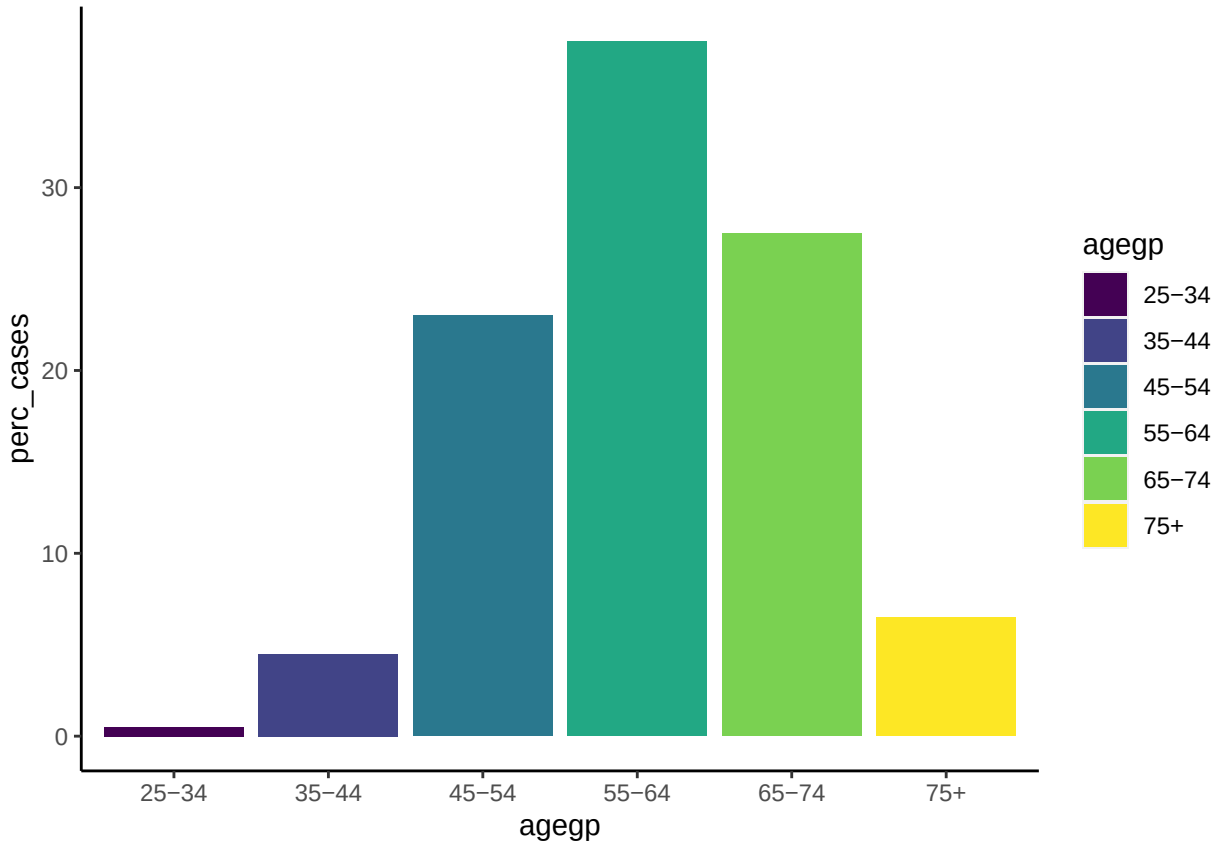
```
# Remove Vertical Lines
barchart2 +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank())
```




```
# Remove Horizontal Lines
barchart2 +
    theme(panel.grid.major.y = element_blank(),
          panel.grid.minor.y = element_blank())
```

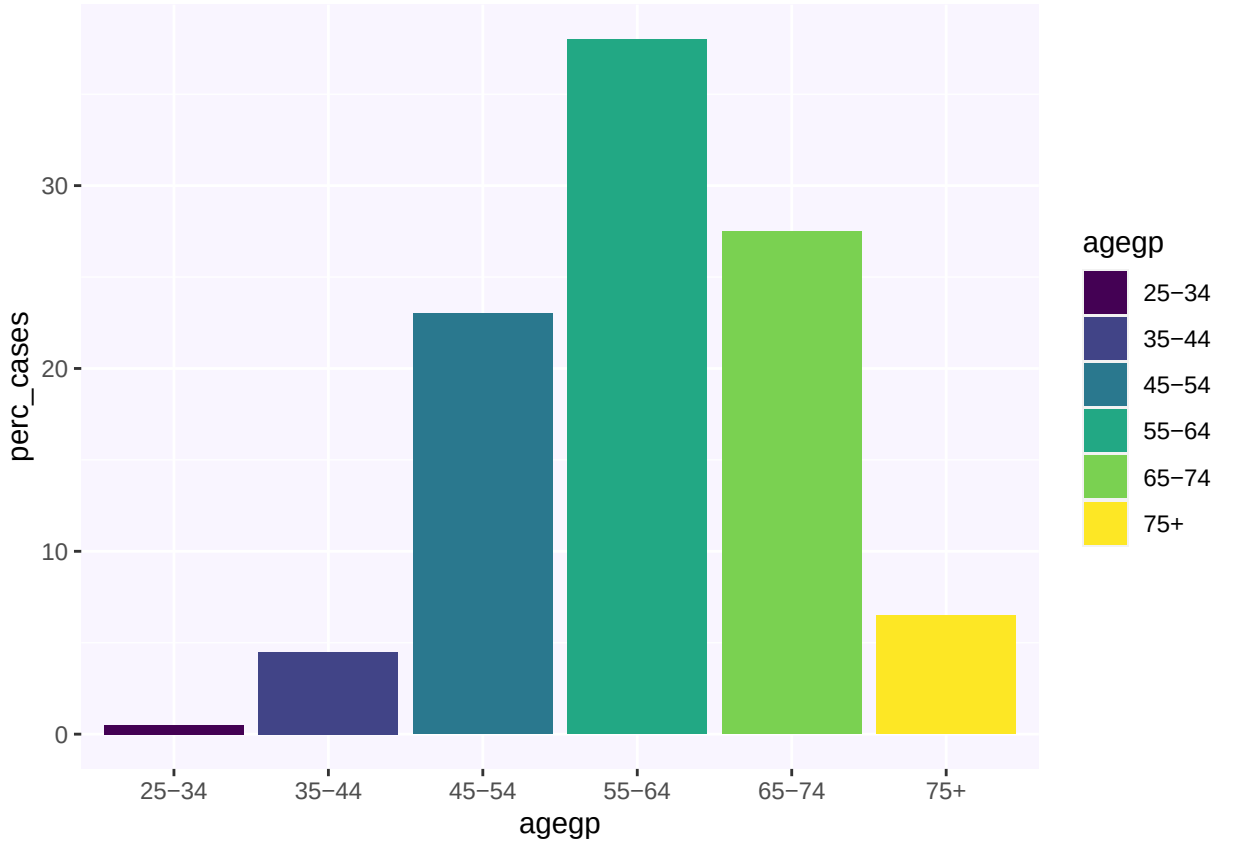



```
# Remove both the lines and background
barchart2 +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.line = element_line(colour = "black"))
```

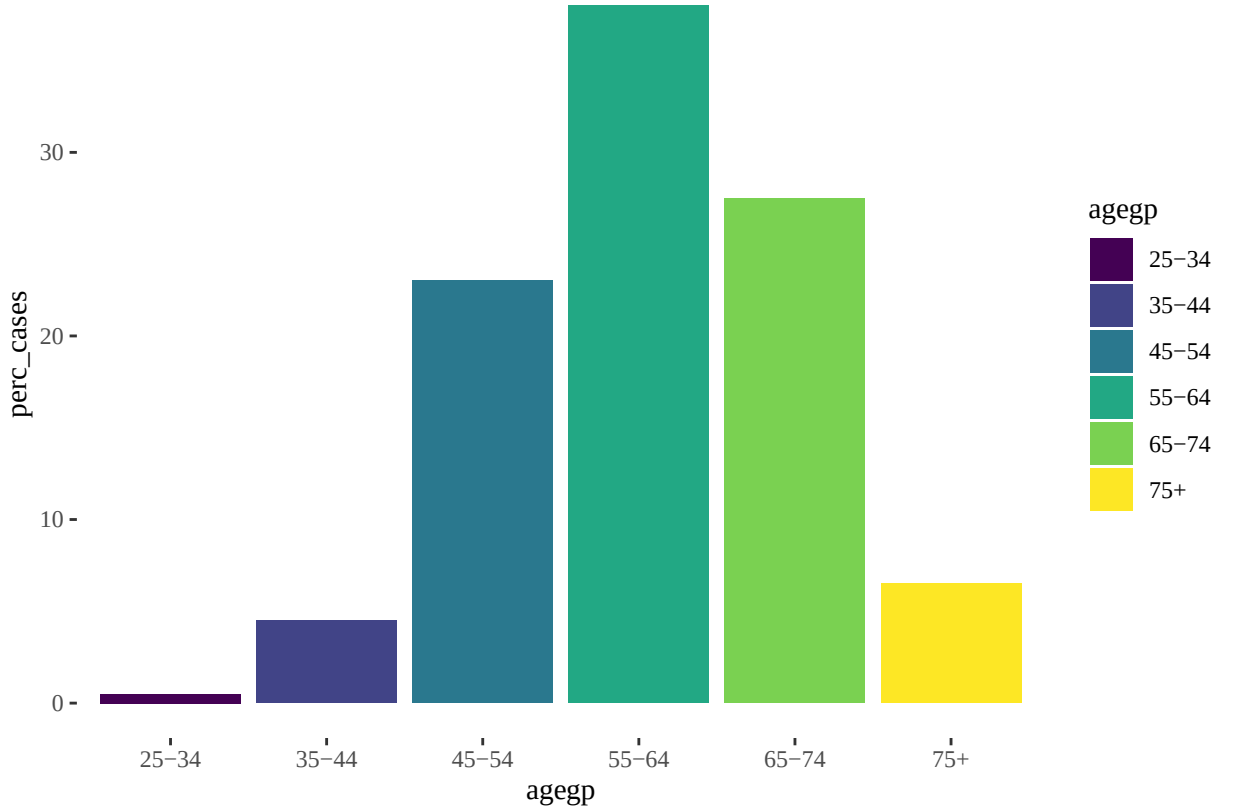



```
# Add Custom background color
barchart2 +
    theme(panel.background = element_rect(fill = "#F9F5FF",
                                          size = 2, linetype = "solid"))
```

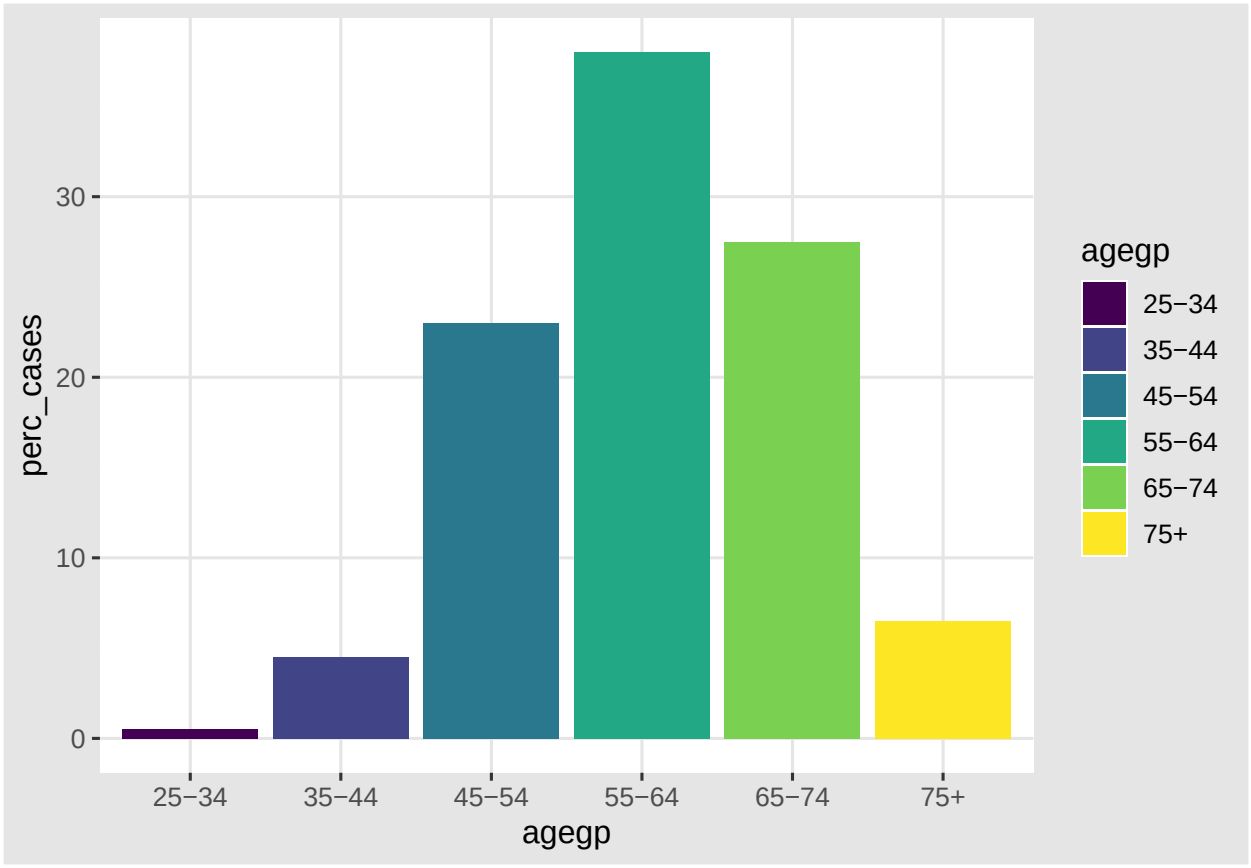
```
## Warning: The ‘size‘ argument of ‘element_rect()‘ is deprecated as of ggplot2 3.4.0.
## i Please use the ‘linewidth‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



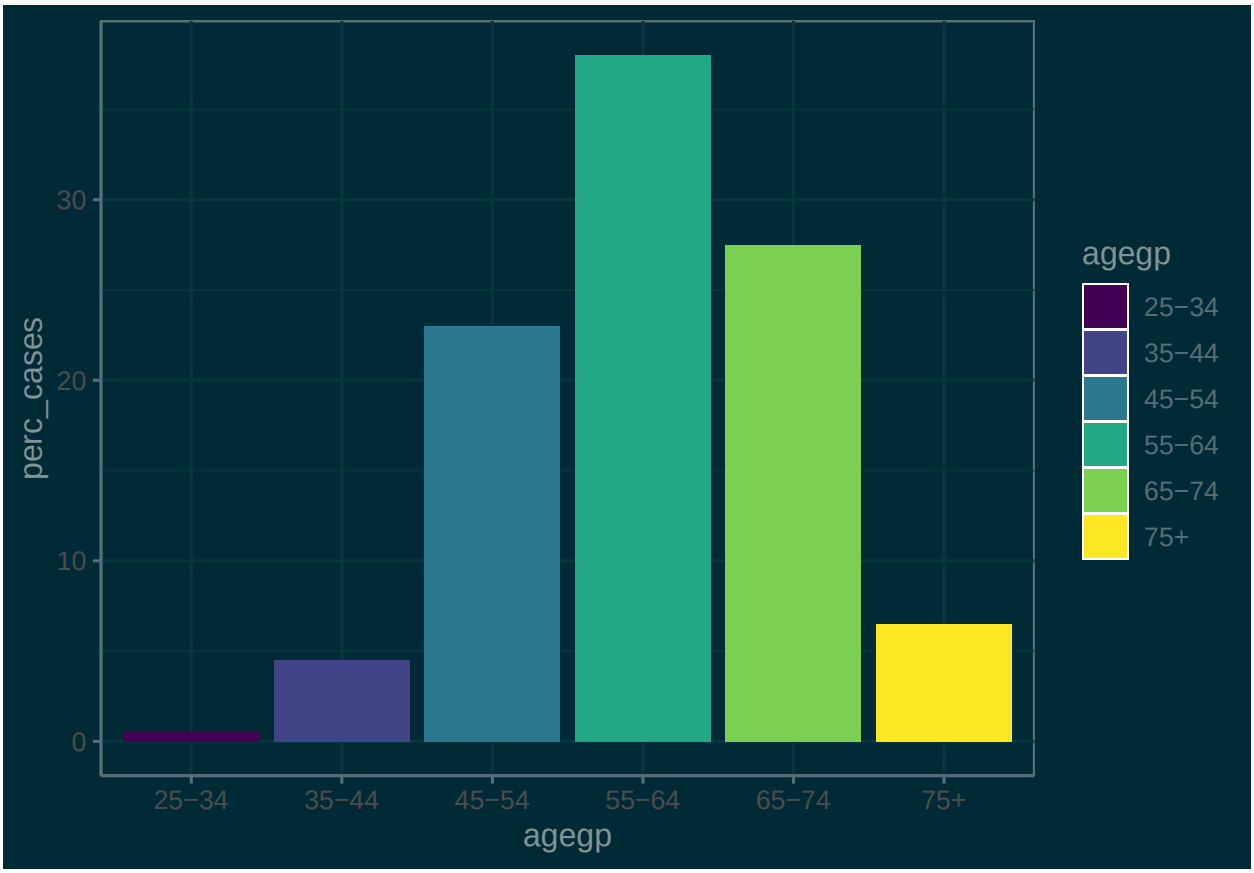

```
# Use ggthemes to add/remove grid lines, change color of background, and change plot themes

# Example 1: Minimalist Theme
barchart2 +
  theme_tufte()
```




```
# Example 2: Inverse Gray Theme
barchart2 +
  theme_igray()
```




```
# Example 3: Dark Theme
barchart2 +
  theme_solarized(light = FALSE)
```


perc_cases
30
20
10
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
agegp
25−34
35−44
45−54
55−64
65−74
75+

Part 3: Types of bar charts

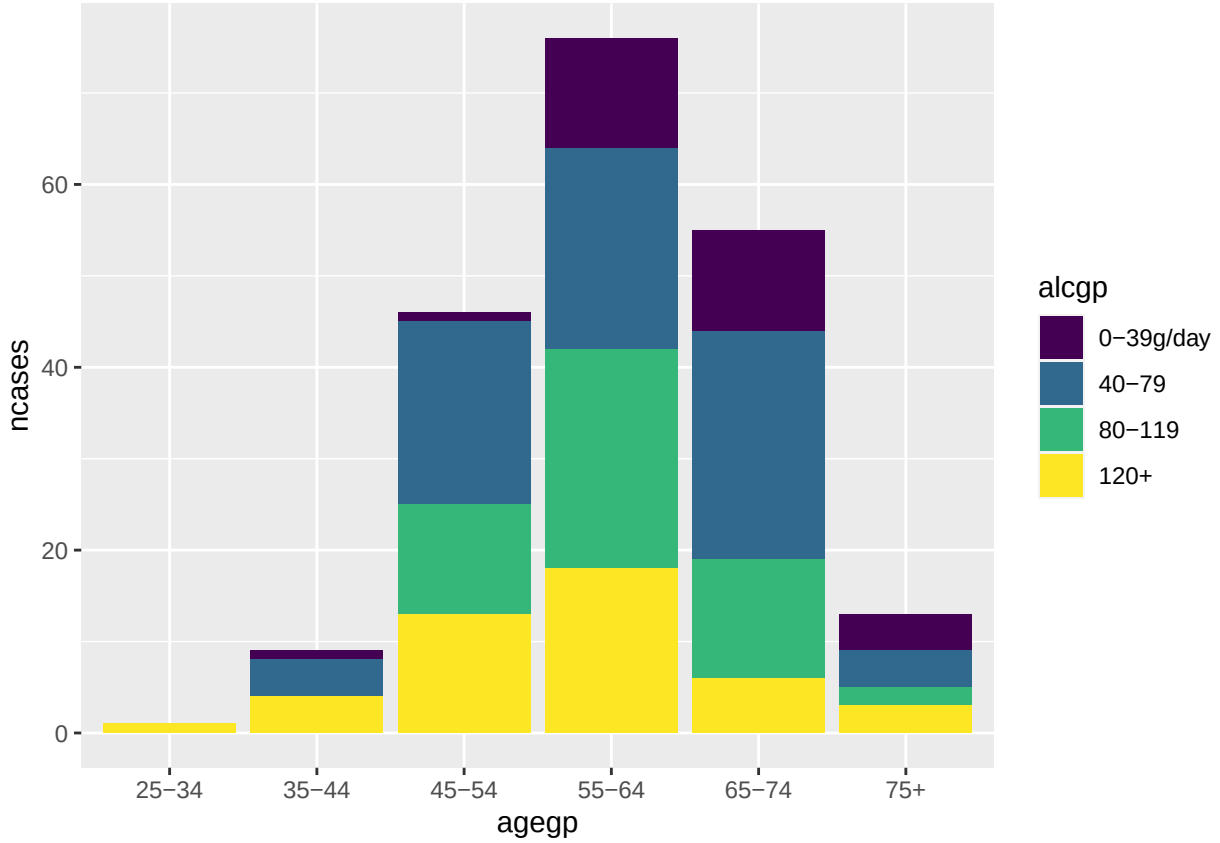
Alright, so we have explored how to modify regular bar charts to make them more professional and presentable.

In this part of the module, we will dive into creating more complex bar charts such as grouped and stacked bar charts.

3.1: Stacked bar chart

Earlier, we compared the number and percent of cases of esophageal cancer by age group and noticed that some age groups tended to have a higher percentage of cases than others. To explore this relationship more, let's look at how the number of cases are broken down by each age group's alcohol consumption.

```
stacked_bc <- ggplot(esoph, aes(x = agegp, y = ncases, fill = alcgp)) +
    geom_bar(stat = "identity")

stacked_bc
```

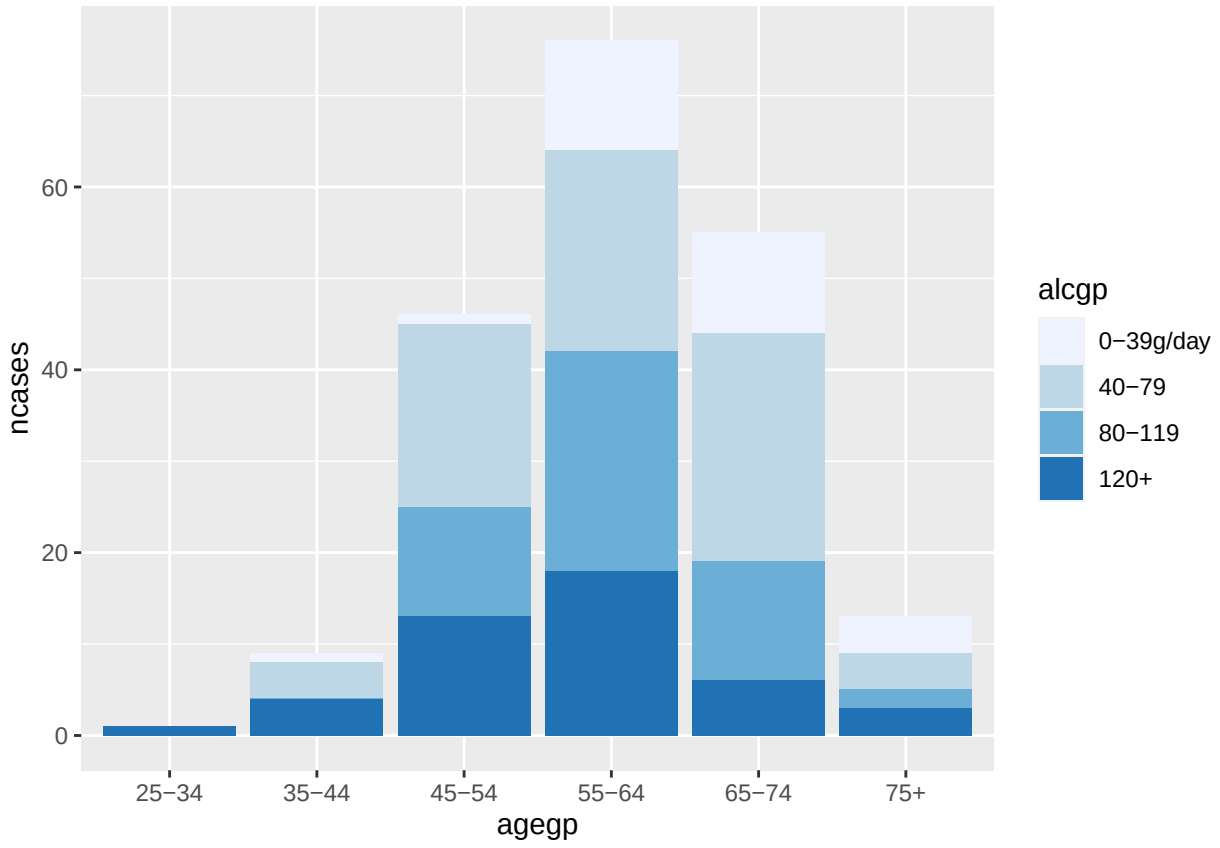


What do you notice about the distribution of alcohol consumption across the age groups?

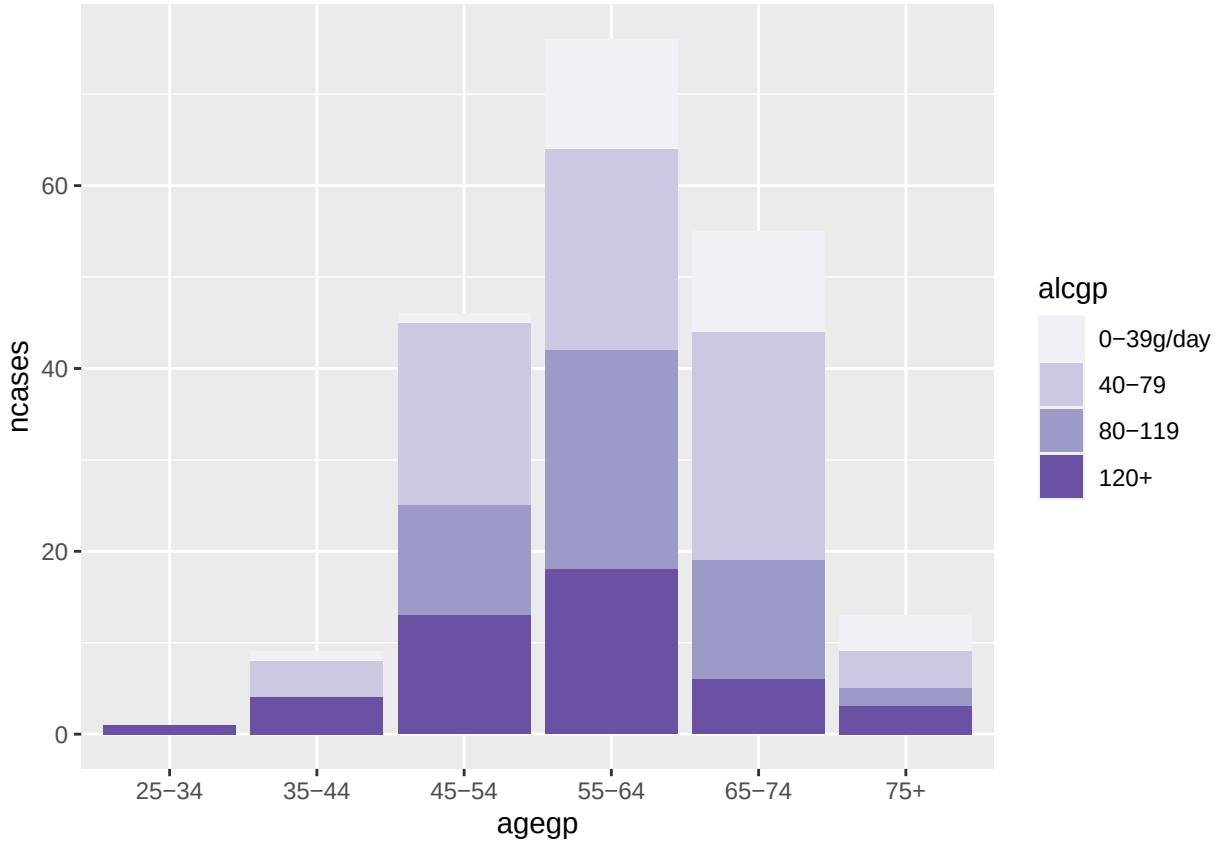
Currently, our stacked bar chart is using the default colors which may or may not look great or fit with our theme.

3.1.1: Color

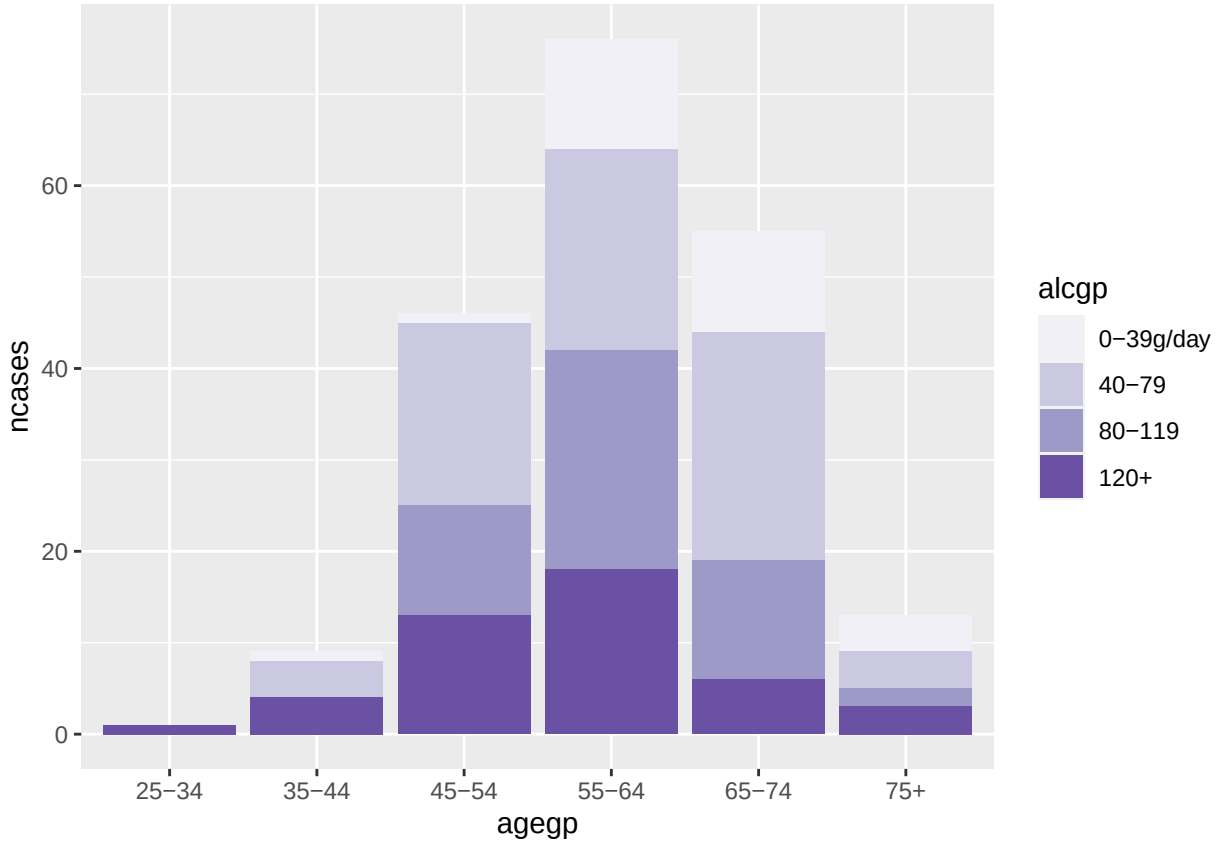
```
# Color Palettes
stacked_bc + scale_fill_brewer() #Default Brewer color palette
```



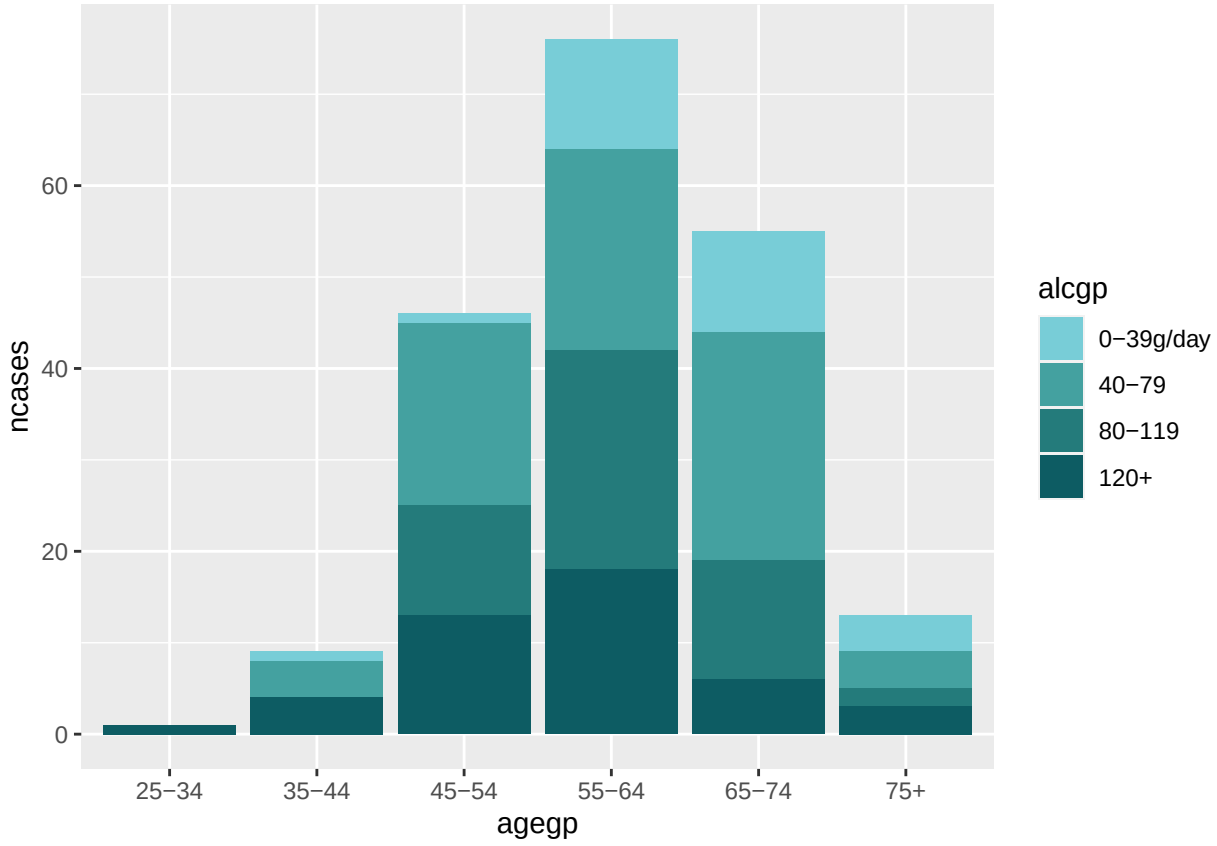

```
# Can specify the palette by palette name or number
stacked_bc + scale_fill_brewer(palette = 12) #OR
```




```
stacked_bc + scale_fill_brewer(palette = "Purples")
```

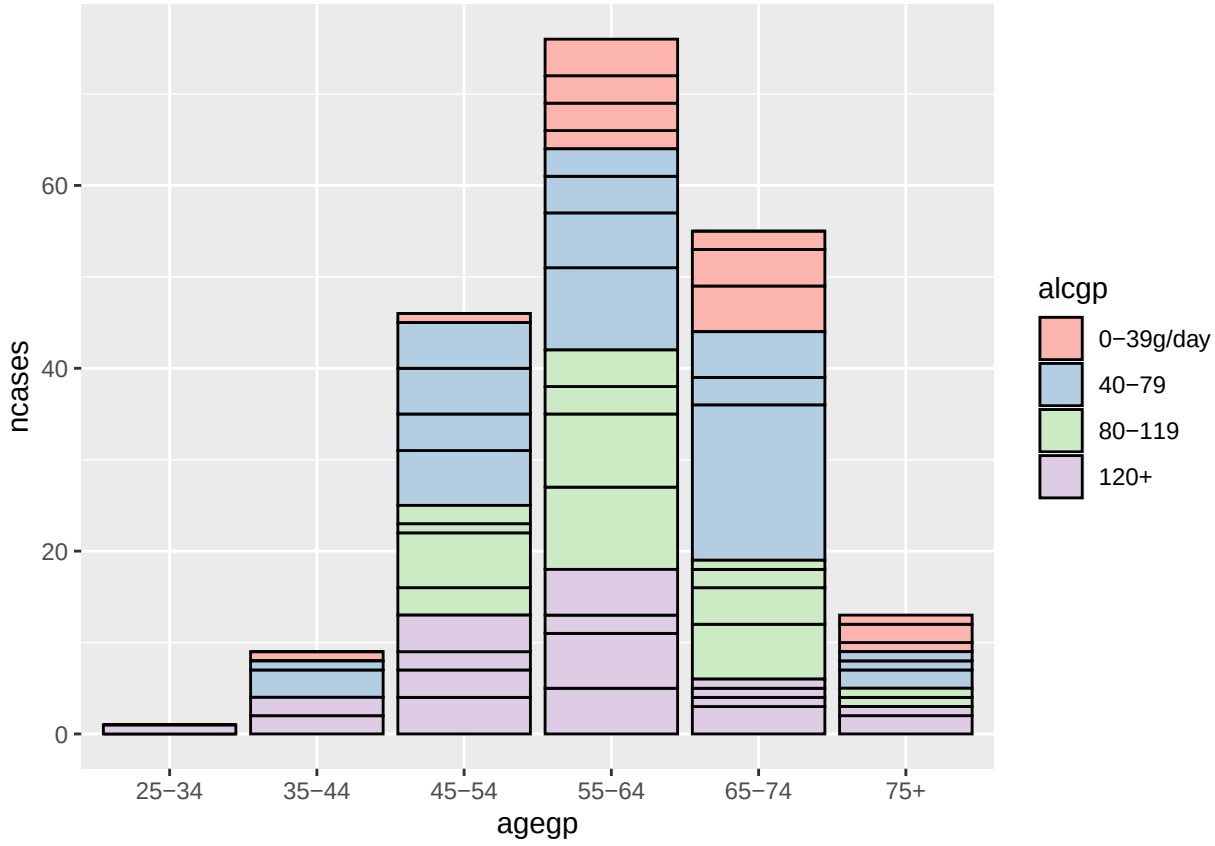



```
# Manually Choose Colors for what you are stratifying by (i.e. alcgp)
stacked_bc + scale_fill_manual(values=c("#78CDD7", "#44A1A0", "#247B7B", "#0D5C63"))
```


ncases
60
40
20
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
alcgp
0−39g/day
40−79
80−119
120+

3.1.2: Border

```
# Add Border and Color
ggplot(esoph, aes(x = agegp, y = ncases, fill = alcgp)) +
  geom_bar(stat = "identity", color = "black") + #Specify Border by "color ="
  scale_fill_brewer(palette = "Pastel1")
```

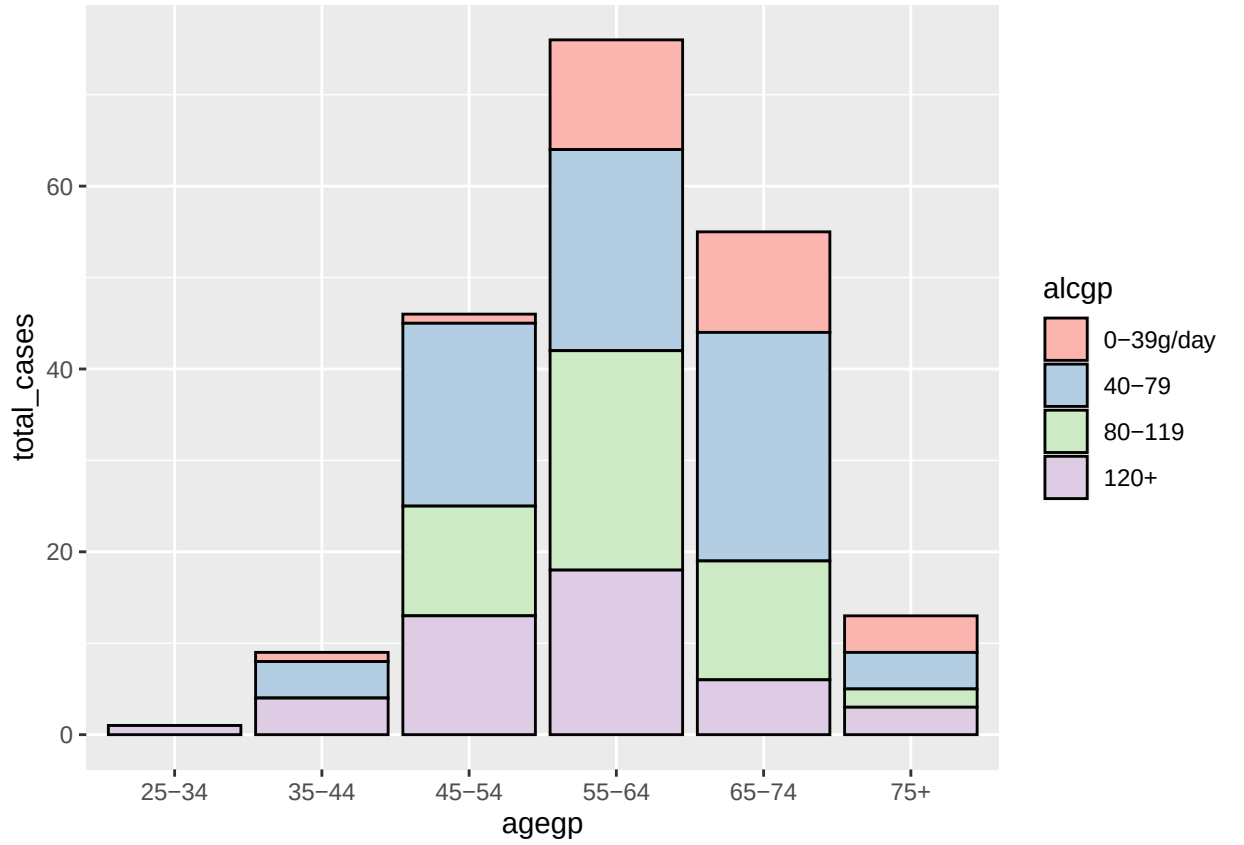



```
# The borders look a bit weird due to the way the data is represented in our dataset
# (We have the same alcgp category duplicated multiple times for each agegp)

# To get the proper border, we will need to clean our data such that
# the alcgp category is not repeated for each agegp
alc_cases <- esoph %>%
  select(agegp, alcgp, ncases) %>%
  group_by(agegp, alcgp) %>%
  summarize(total_cases = sum(ncases))
```

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

```
# Compare this new clean data with the original dataset.
# What do you notice about the ways agegp, alcgp, and ncases are represented?

barchart3 <- ggplot(alc_cases, aes(x = agegp, y = total_cases, fill = alcgp)) +
```

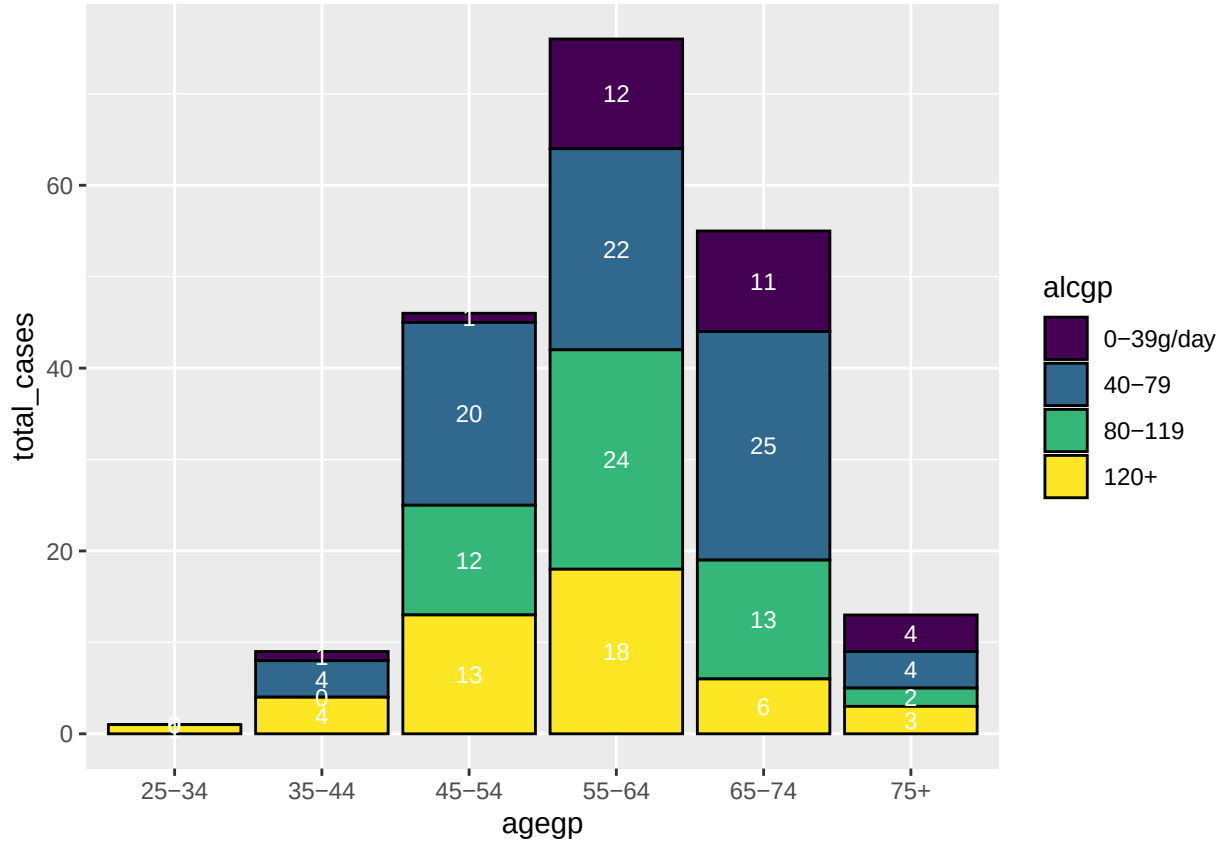
```
  geom_bar(stat = "identity",color = "black") +
  scale_fill_brewer(palette = "Pastel1")

barchart3
```



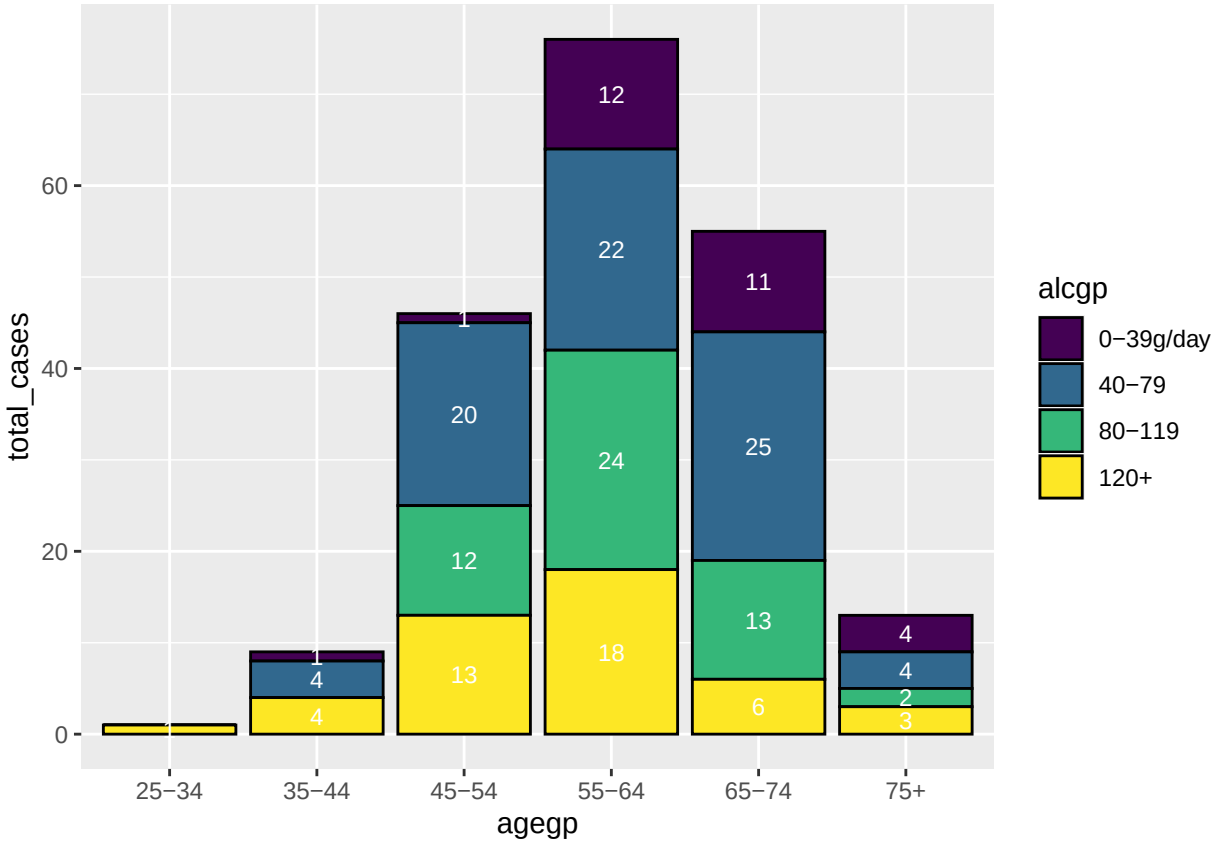

```
# Now the borders look good!
```

3.1.3: Labels

```
ggplot(alc_cases, aes(x = agegp, y = total_cases, fill = alcgp,
                      label = total_cases)) + # Need to specify the label
  geom_bar(stat = "identity", color= "black") +
  geom_text(position = position_stack(vjust = 0.5), size = 3, color = "#ffffff")
```

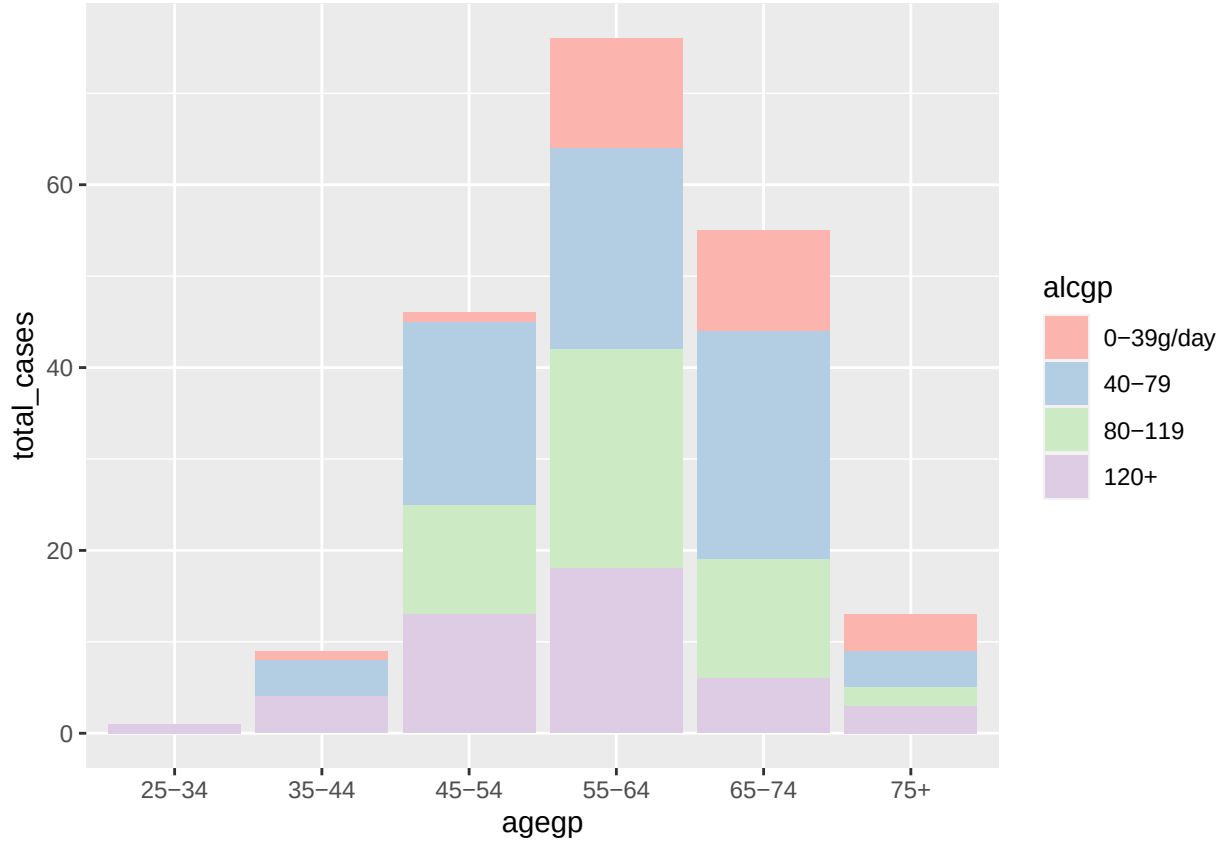



```
# need to mention position = position_stack(); vjust = 0.5 is for centering

# In this case, this barchart is showing the values of 0, which we don't want,
# so we can modify our code
ggplot(alc_cases, aes(x=agegp, y=total_cases, fill = alcgp,
                      label = total_cases)) +
  geom_bar(stat="identity", color= "black")+
  geom_text(data=subset(alc_cases, total_cases != 0),
            position = position_stack(vjust = 0.5), size=3, color = "#ffffff")
```


total_cases
60
40
20
0
12
22
11
1
20
25
24
12
13
18
4
1
4
4
13
6
4
4
2
3
1
25−34
35−44
45−54
55−64
65−74
75+
agegp
alcgp
0−39g/day
40−79
80−119
120+

3.1.4: Legend

```
# Change the Category Labels
ggplot(alc_cases, aes(x = agegp, y = total_cases, fill = alcgp)) +
  geom_bar(stat = "identity") +
  scale_fill_brewer(palette = "Pastel1")
```

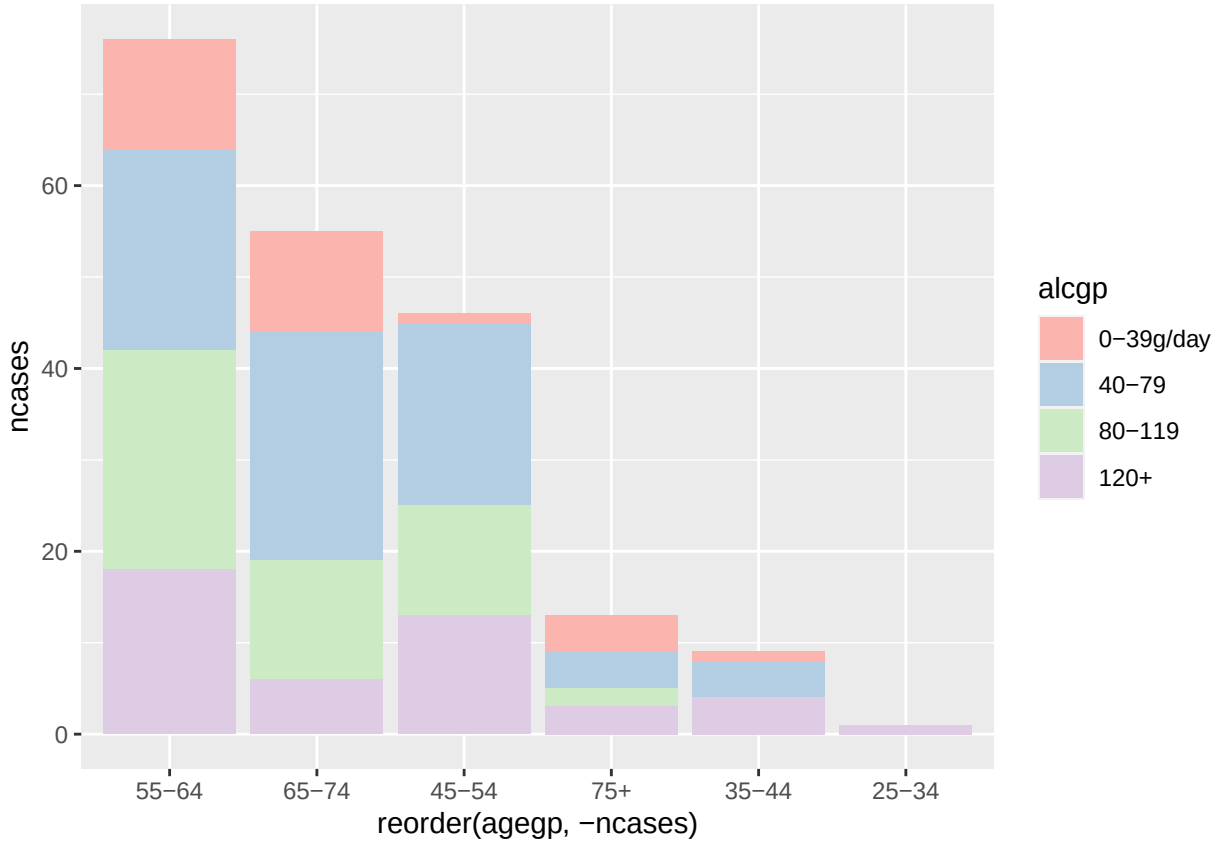



```
  guides(fill=guide_legend( # 'Guide legend' allows to manually input a legend
    title="Alcohol Use \n    (g/day)")) + # '\n' adds a new line and then add spaces to center '(g/day)
  scale_fill_discrete(labels = c("<= 39", "40-79", "80-119", ">= 120"))
```

```
## NULL
```

3.1.5: Ordering Bars

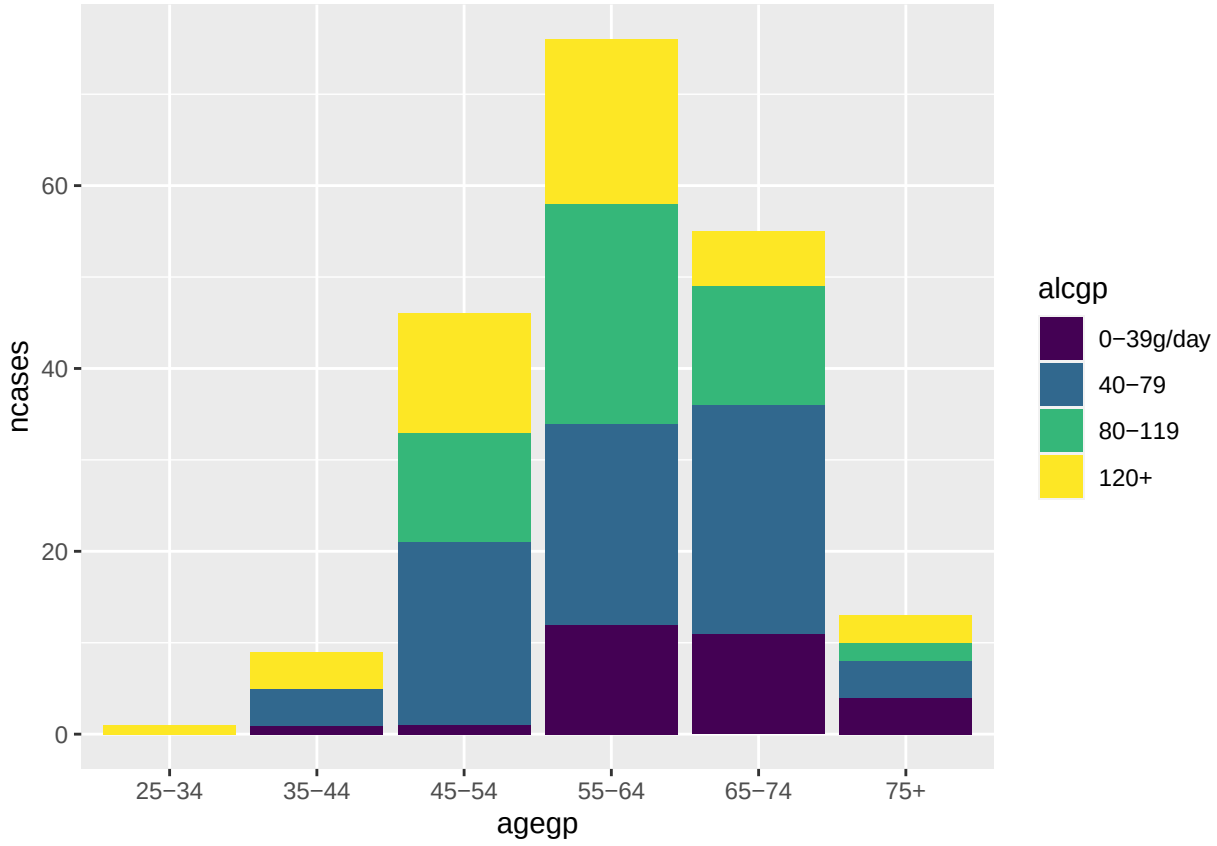
```
# Reordering the bars in ascending/descending order
ggplot(esoph,aes(x = reorder(agegp,-ncases), y = ncases, fill = alcgp)) +
  geom_bar(stat ="identity") +
  scale_fill_brewer(palette = "Pastel1")
```




```
# Syntax: x = reorder(X variable,+/-Y variable); + = ascending, - = descending
# NOTE: It does not make sense to reorder the bars in this context
# as the age group categories are out of order
```

3.1.6: Reverse bar stacking

```
# Reverse the stacking of the bars
ggplot(esoph,aes(x = agegp, y = ncases, fill = alcgp)) +
  geom_bar(stat ="identity", position = position_stack(reverse = TRUE)) # Reversing stacking
```

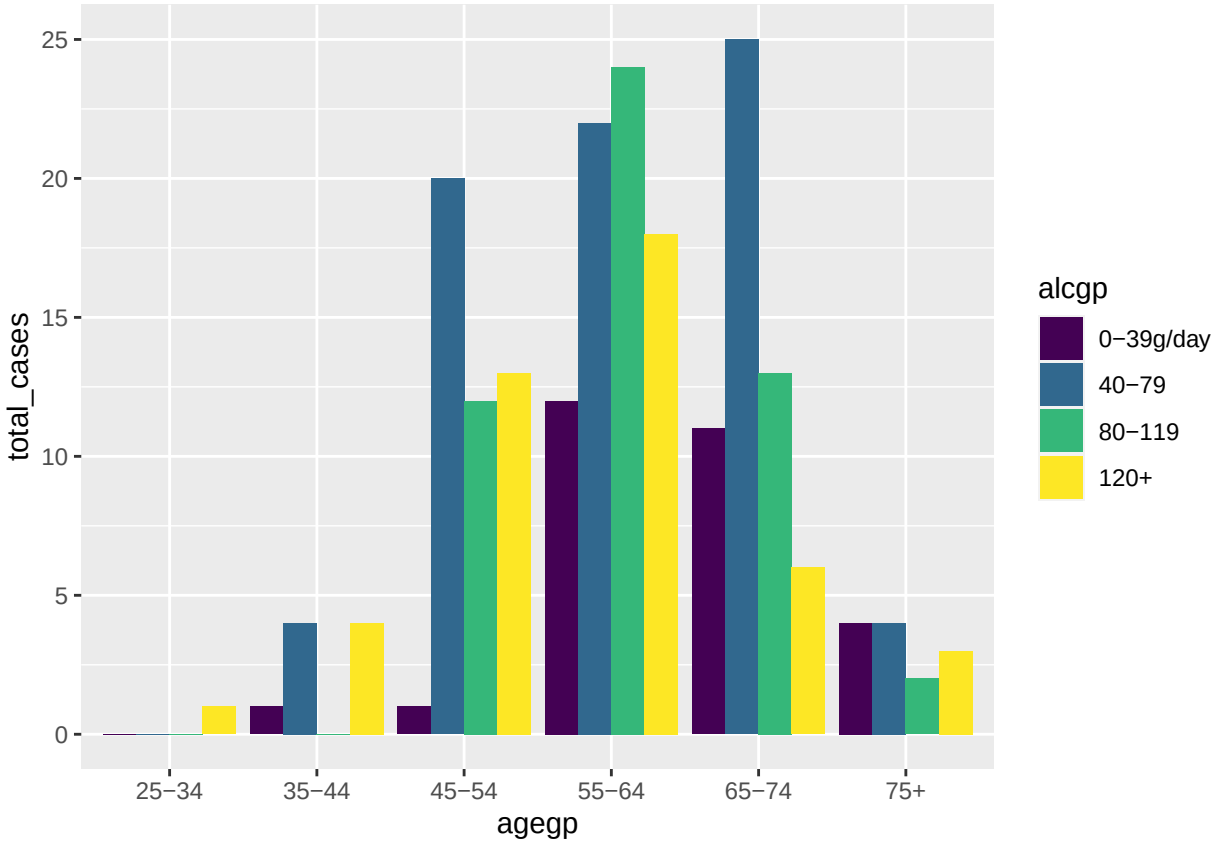

ncases
60
40
20
0
25−34
35−44
45−54
55−64
65−74
75+
agegp
alcgp
0−39g/day
40−79
80−119
120+

3.2: Dodged bar charts

Sometimes a stacked barchart may not be easy to understand or interpret. Well, there is a solution to that: Dodged bar charts!

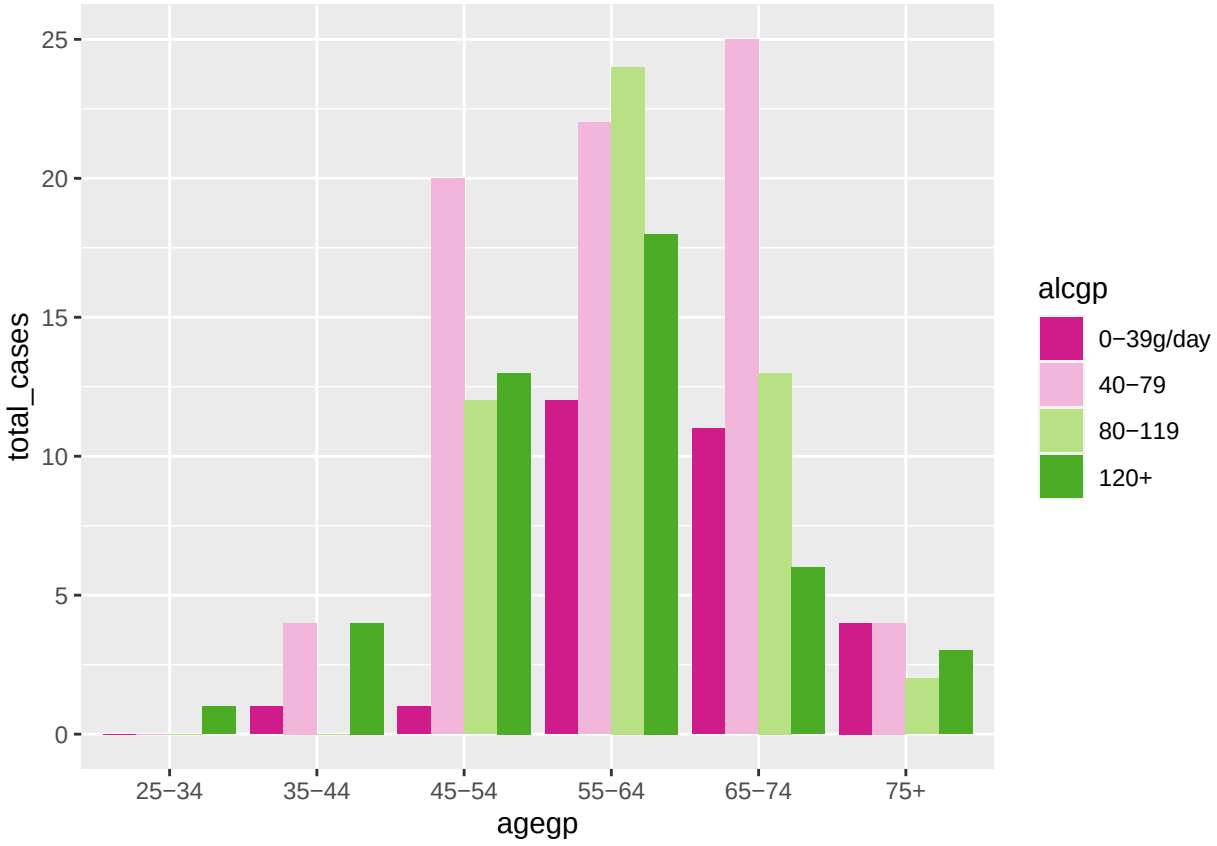
Dodged bar charts are very similar to stacked bar charts, with very minor changes in code syntax. We will look at grouped bar charts with our previous example of alcohol consumption.

```
dodged_bc <- ggplot(alc_cases, aes(x = agegp, y = total_cases, fill = alcgp)) +
  geom_bar(stat = "identity",
           position = "dodge")  # Need to specify position = "dodge" to group bars next to each other
dodged_bc
```



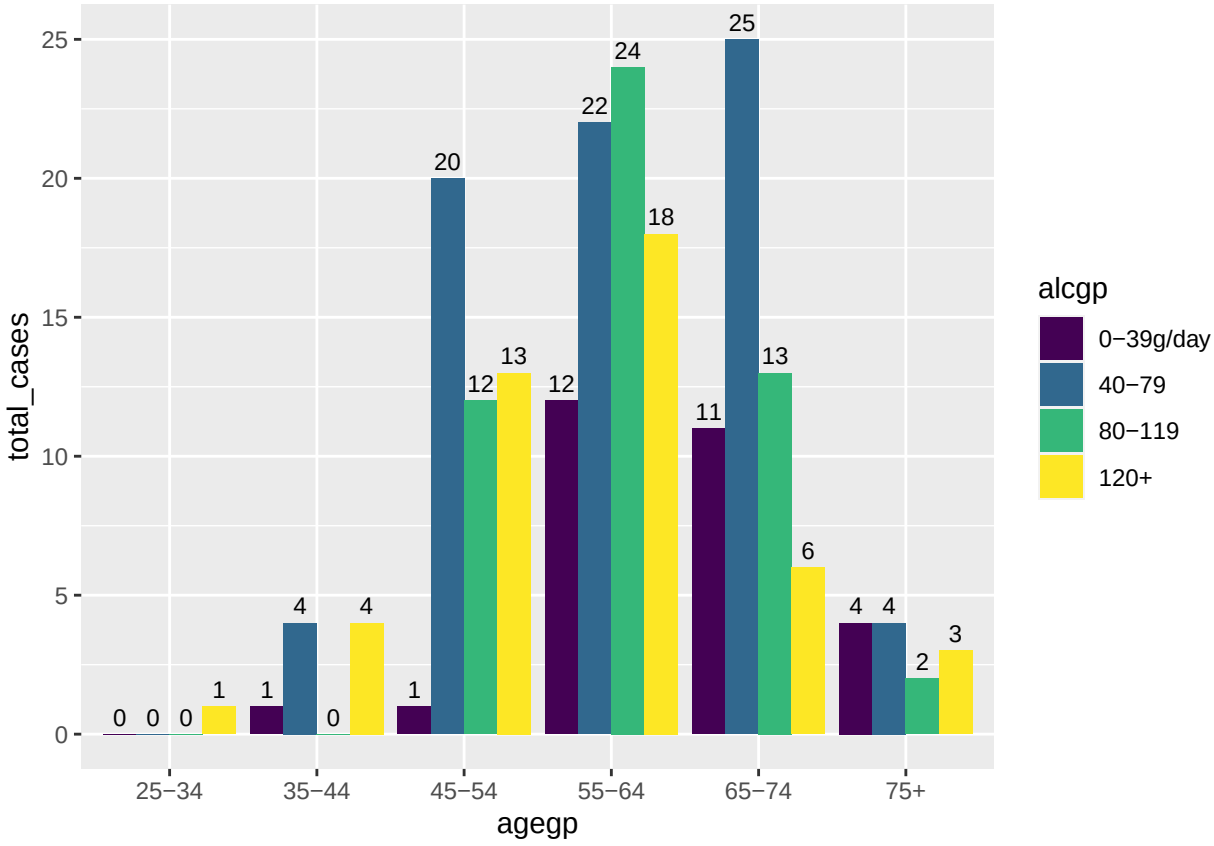
3.2.1: Color

```
# Change color the same way you did for stacked barcharts
# with either scale_fill_manual or scale_fill_brewer

dodged_bc + scale_fill_brewer(palette = "PiYG")
```

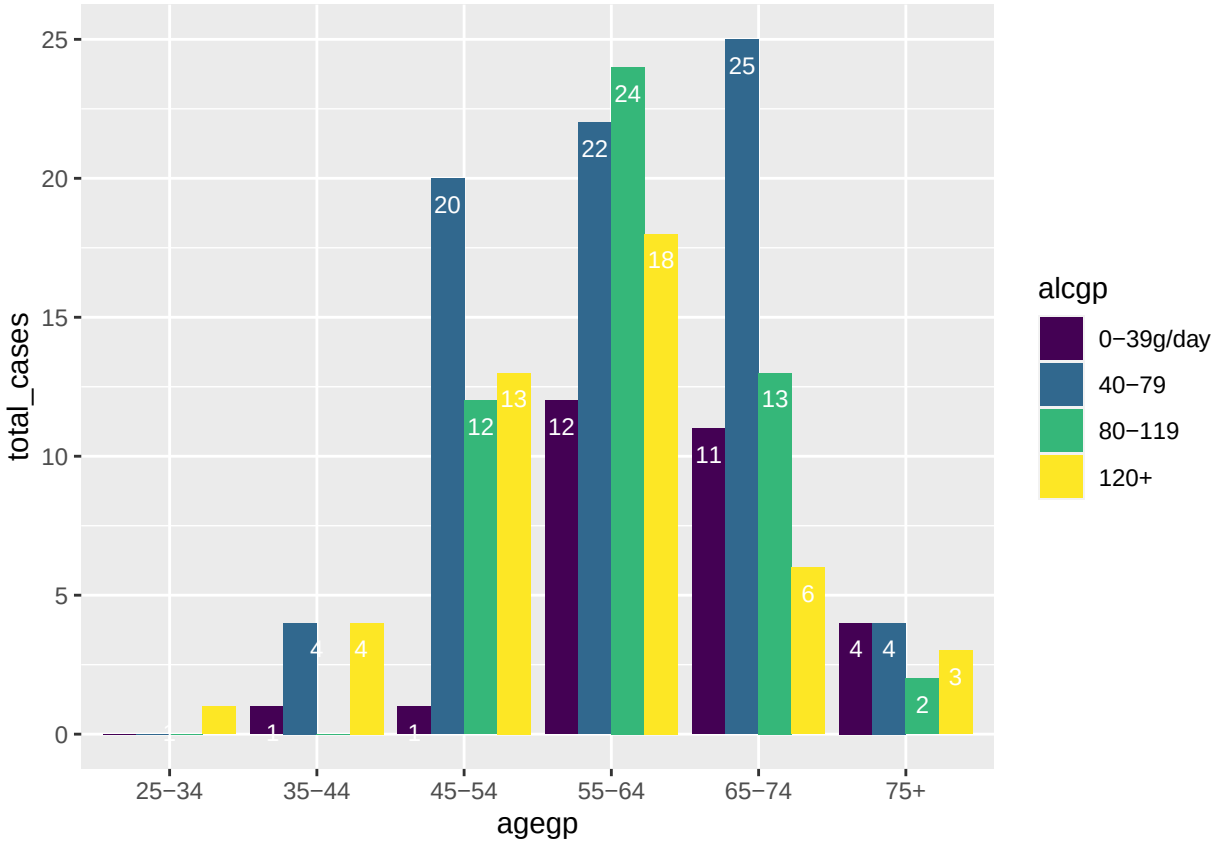


3.2.2: Labels

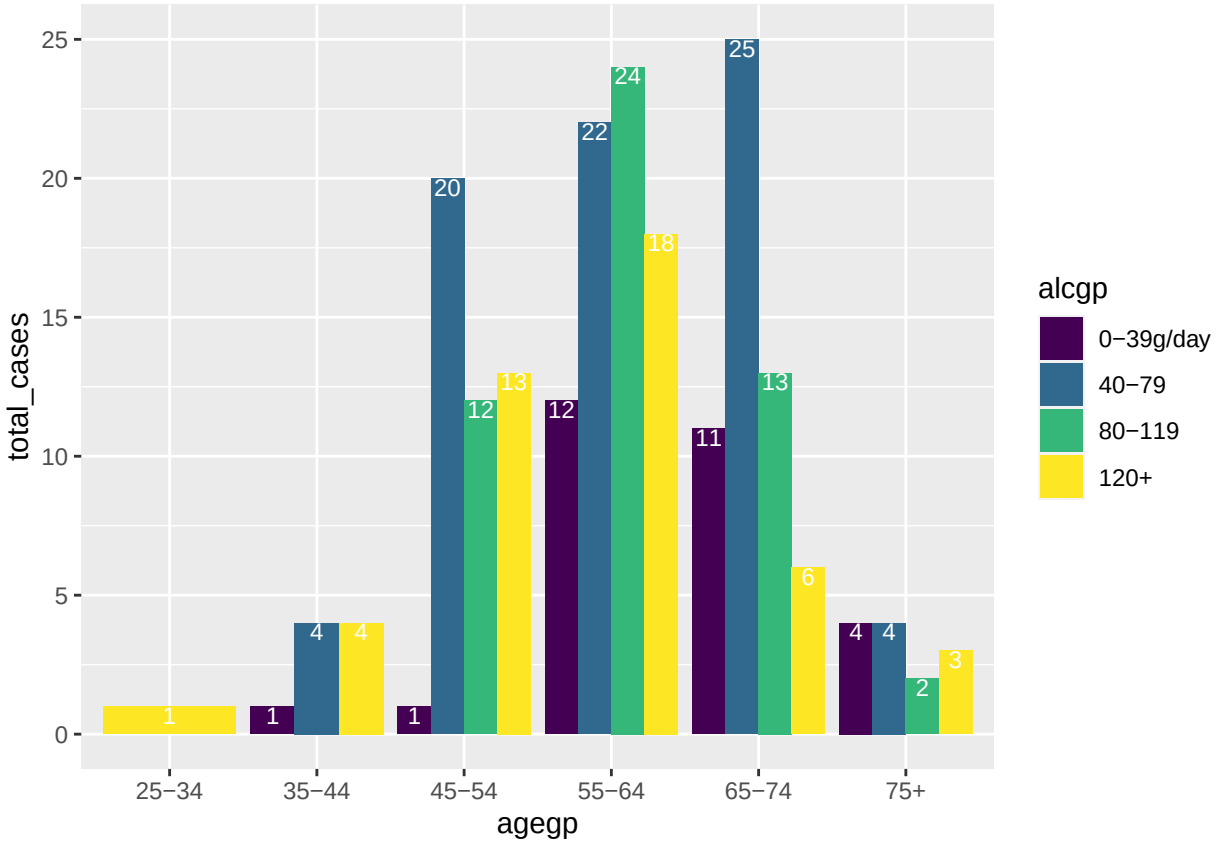
```
# If you would like to include the empty bar space along with the values
dodged_bc +
  geom_text(aes(label = total_cases), position = position_dodge(0.9),
            vjust = -0.5, size = 3, color = "black")
```




```
# If you do not want values of 0 to show up on the graph
dodged_bc +
  geom_text(data=subset(alc_cases, total_cases != 0),
            aes(label = total_cases), position = position_dodge(0.9),
            vjust = 2, size = 3, color = "#ffffff")
```




```
# Notice that this graph above shows the empty bar space that we don't want and
# therefore some of the numbers are not formatted on the bar.
# To fix this, we need to remove any rows where the total_cases is 0
# from our dataset and plot again

alc_cases2 <- alc_cases %>%
  filter(total_cases > 0)

ggplot(alc_cases2, aes(x = agegp, y = total_cases, fill = alcgp)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = total_cases), position = position_dodge(0.9),
            vjust = 1, size = 3, color = "#ffffff") # Much better!
```



The code for everything else for dodged bar charts is similar to what has been previously covered in this module.

3.3: Faceted bar charts

3.3.0: Data Import

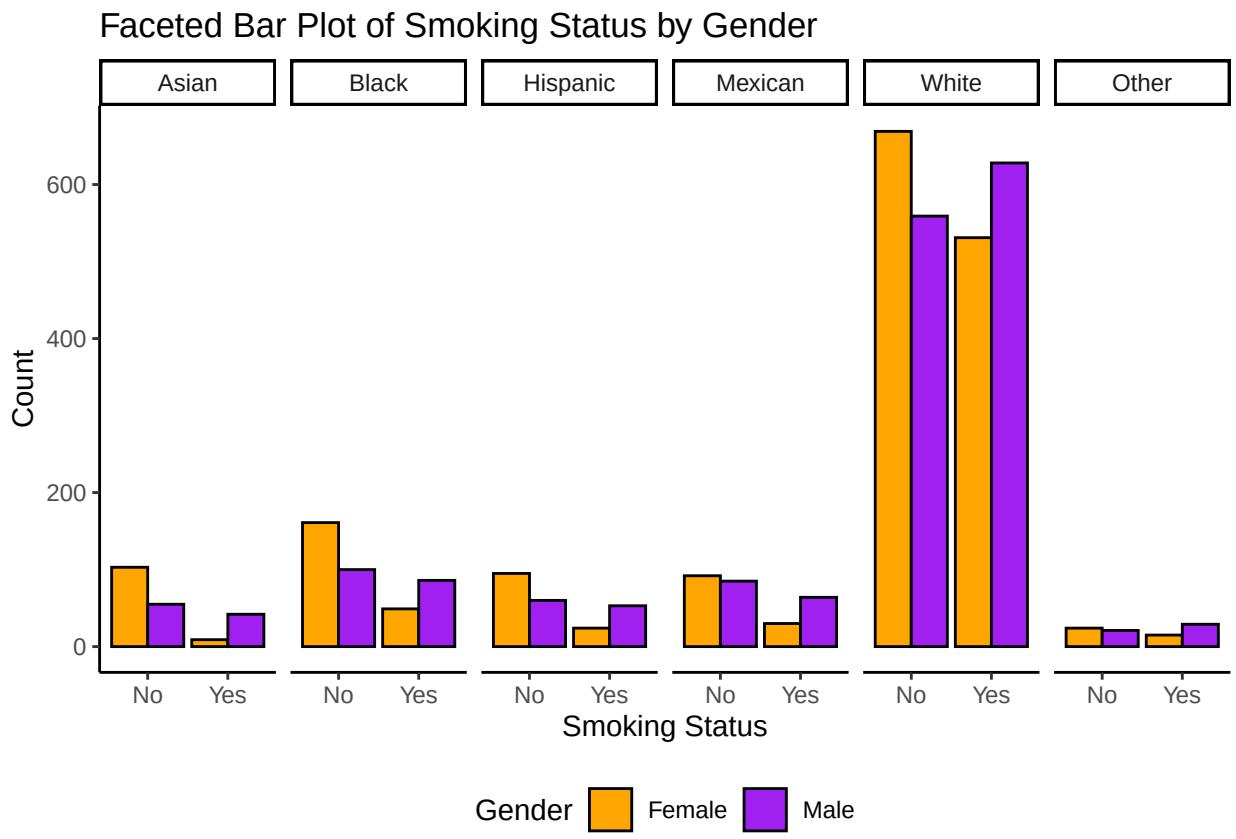
The **esoph** data set is too small and has too few features to support quality faceting therefore we will be leveraging the **NHANES** database to demonstrate **faceted bar charts**.

```
# Install the NHANES package if you need to install it
# install.packages("NHANES")

# Load NHANES library
library(NHANES)

# Load the "NHANES" dataset
data(NHANES)

# Dropping NA values
NHANES_df <- NHANES %>%
  drop_na(Race3,Education)
```

3.3.1: Method 1: facet_grid()

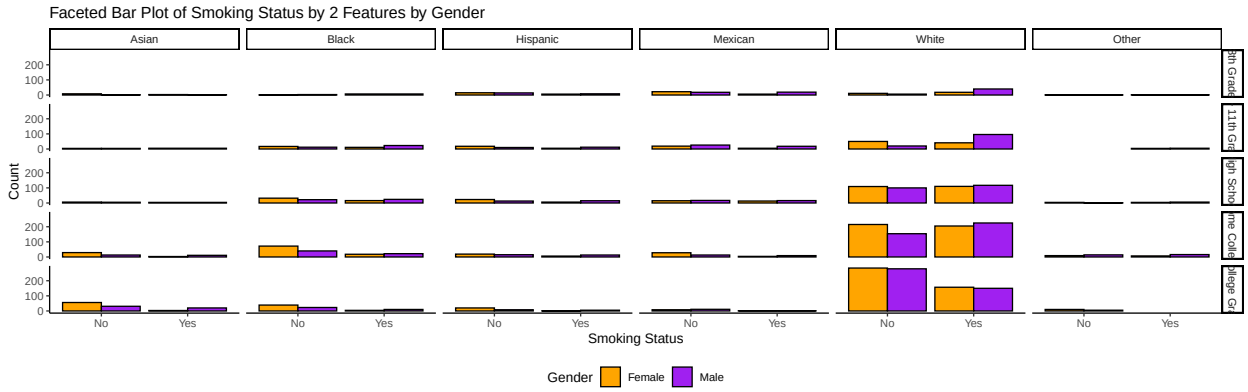
```


# Making it look cleaner
ggplot(NHANES_df, aes(x = Smoke100, fill = Gender)) +
  geom_bar(position = "dodge", color = "black") +
  labs(x = "Smoking Status", y = "Count", title = "Faceted Bar Plot of Smoking Status by Gender") +
  scale_fill_manual(values = c("male" = "purple", "female" = "orange"),
                    labels = c("Female", "Male")
                    ) + # Be mindful of assigning colors to gender
  facet_grid(~ Race3) + # 1 Feature: Education by Race
  theme_classic() +
  theme(legend.position = "bottom")
```

3.3.1.1: Faceting by one feature



3.3.1.2: Faceting by two features

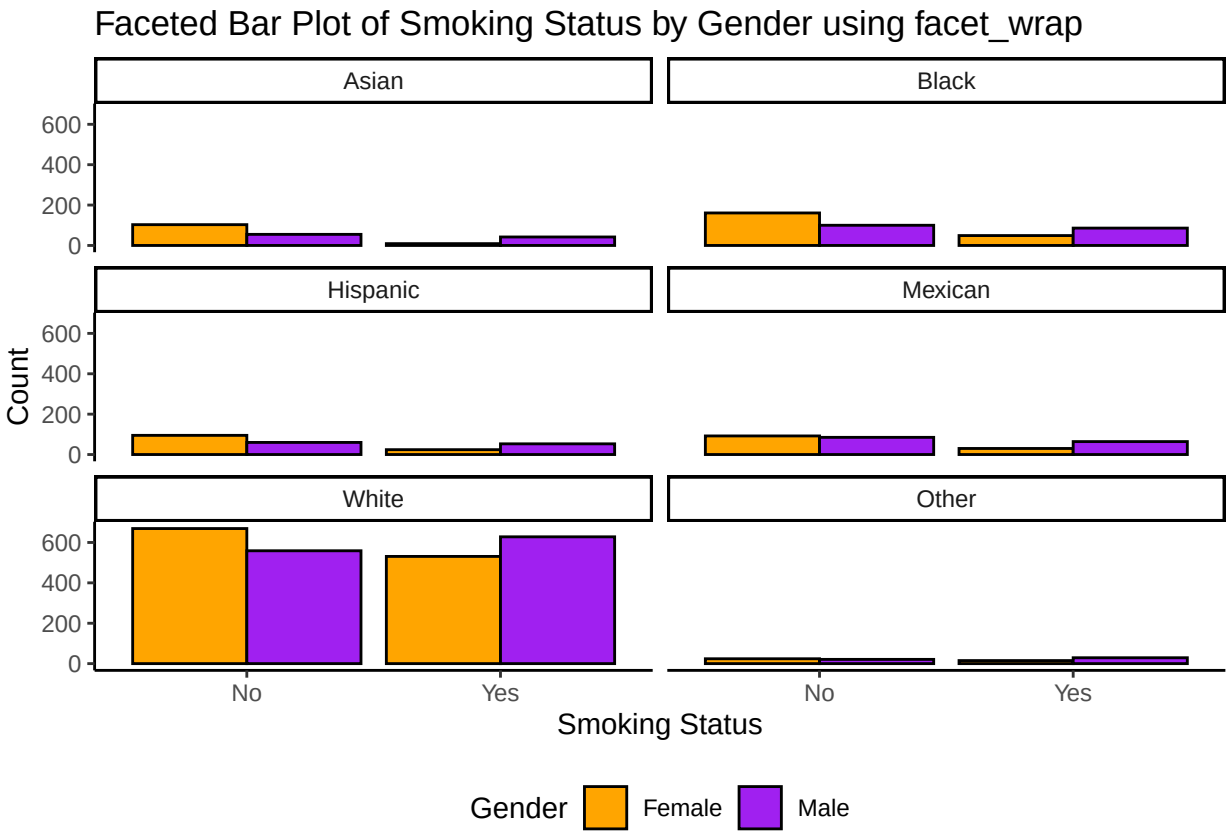
```
ggplot(NHANES_df, aes(x = Smoke100, fill = Gender)) +
  geom_bar(position = "dodge", color="black") +
  labs(x = "Smoking Status", y = "Count",
       title = "Faceted Bar Plot of Smoking Status by 2 Features by Gender") +
  scale_fill_manual(values = c("male" = "purple", "female" = "orange"),
                    labels = c("Female", "Male")) +
  facet_grid(Education ~ Race3) + # 2 Features: Education by Race
  theme_classic() +
  theme(legend.position = "bottom")
```



3.3.2: Method 2: facet_wrap()

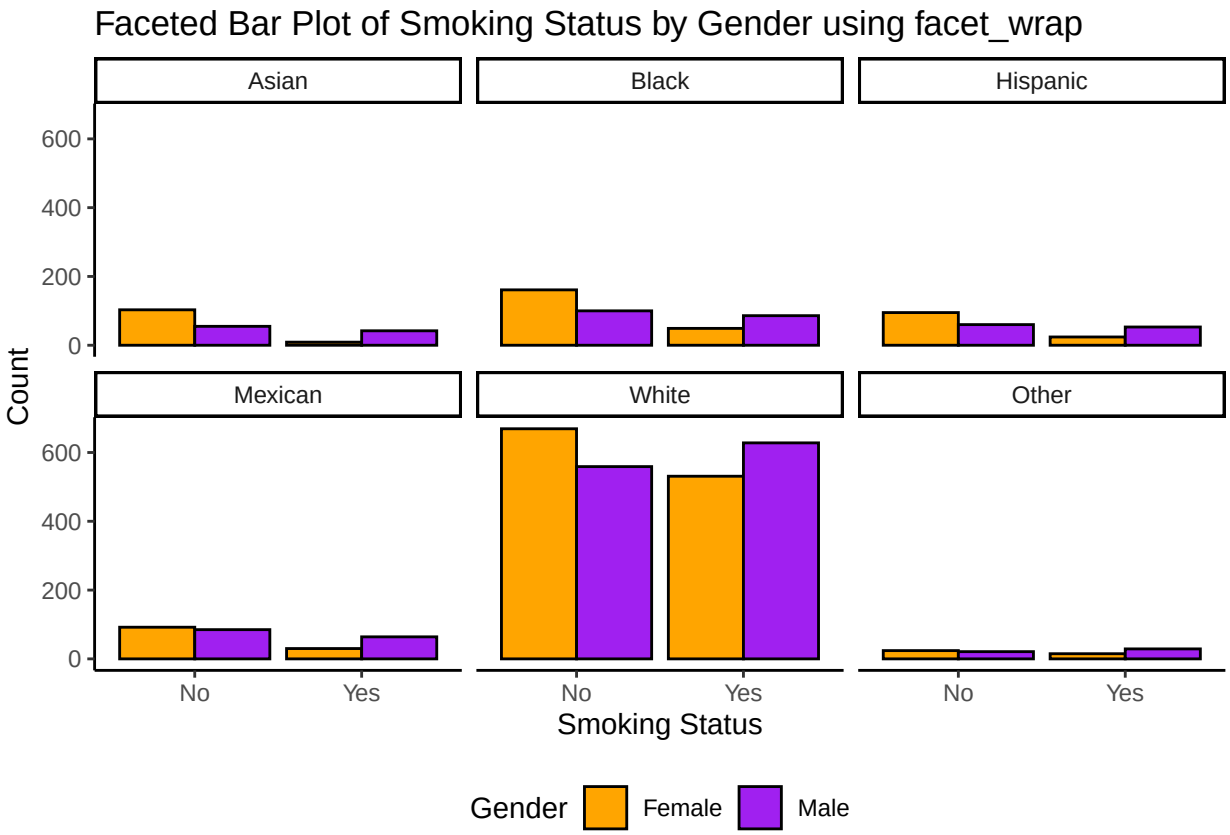
```
# Create a faceted bar plot of smoking status by gender using facet_wrap with 2 columns
ggplot(NHANES_df, aes(x = Smoke100, fill = Gender)) +
  geom_bar(position = "dodge", color= "black") +
  labs(x = "Smoking Status", y = "Count", title = "Faceted Bar Plot of Smoking Status by Gender using fa
  scale_fill_manual(values = c("male" = "purple", "female" = "orange"),
                    labels = c("Female", "Male")) +
  facet_wrap(~ Race3, ncol = 2) + # 2 columns
  theme_classic() +
  theme(legend.position = "bottom")
```

3.3.2.1: Faceting with 2 columns




```
# Create a faceted bar plot of smoking status by gender using facet_wrap
ggplot(NHANES_df, aes(x = Smoke100, fill = Gender)) +
  geom_bar(position = "dodge", color= "black") +
  labs(x = "Smoking Status", y = "Count", title = "Faceted Bar Plot of Smoking Status by Gender using fa
  scale_fill_manual(values = c("male" = "purple", "female" = "orange"),
                    labels = c("Female", "Male")) +
  facet_wrap(~ Race3, ncol = 3) + # 3 columns
  theme_classic() +
  theme(legend.position = "bottom")
```

3.3.2.2: Faceting with 3 columns



3.3.2.3: Faceting but with subgroup proportions Let's take this chart one step further and change the y-axis to be proportions, so that the different groups are truly comparable. To do this, we rely on our `dyplr` toolkit we covered in previous lectures to create a new aggregated dataset.

This is an example of how we sometimes have to transform datasets before we visualize them. While `ggplot` has powerful innards that can take raw datasets, once we start grouping/faceting by multiple variables and want something like proportions, it often is easier to pre-process the data (and sometimes more understandable, from a programming sense)!

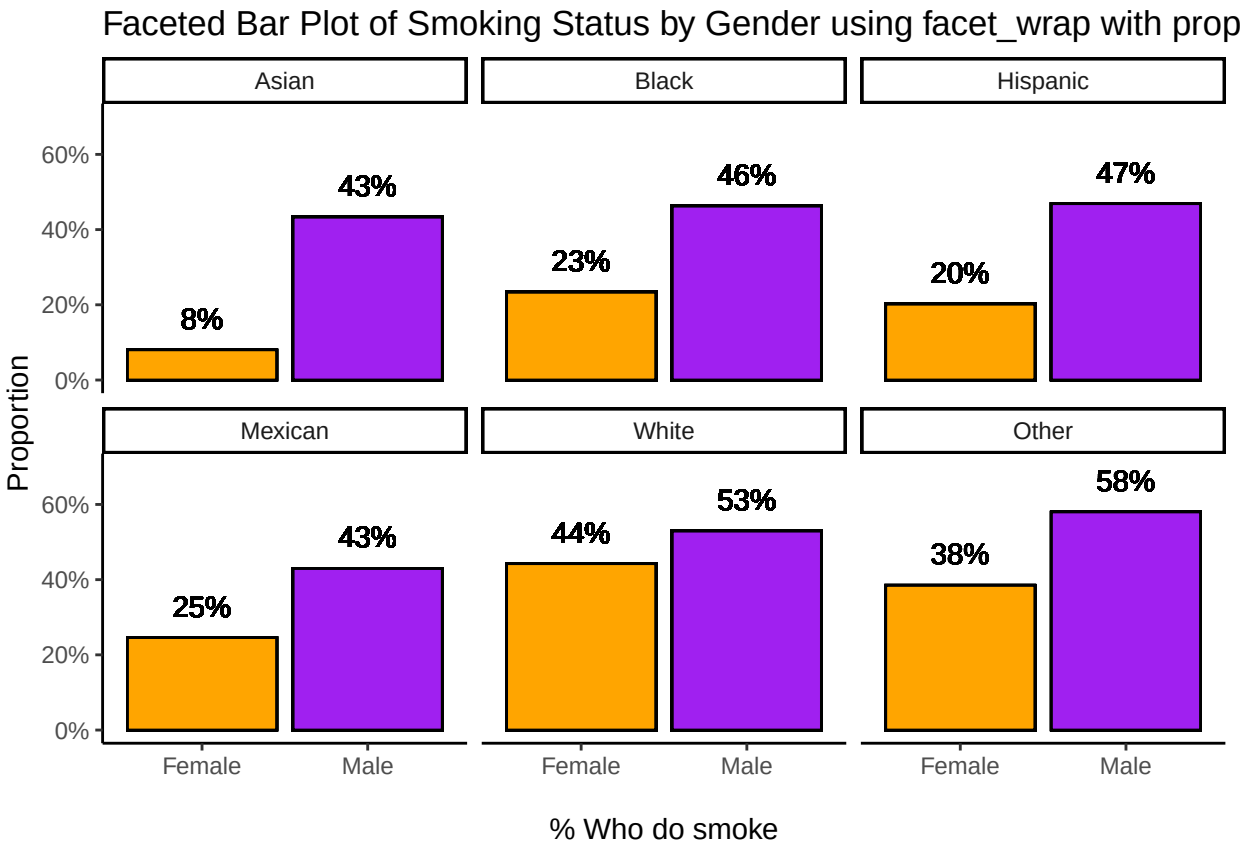
In this subsection, we also make a few aesthetic changes including:

- Adding labels
- Keeping just proportions answering "Yes" (since the absence of Yes is No, and vice versa, therefore could be redundant)
- Use a different way to change labels for Gender
- Piping a newly formed dataframe right into `ggplot`

```
# Create a new aggregated dataset grouped by race3 and gender, describing smoking status

NHANES_df %>%
  # our two grouping variables
  group_by(Gender, Race3) %>%

  # want to get the individual count in each Gender-Race group first
  summarise(Count = n(),
```

```
            # Number of "Yes"es
            Smoke_Yes = sum(Smoke100 == "Yes", na.rm = T),

            # Divide # of "Yes"es by the number of total respondents in a given Gender-Race group
            Freq = Smoke_Yes/Count,

            # Here, we create labels but multiplying Freq, rounding it, and adding "%" to the end
            # This is a character variable, whereas Freq was a numeric!
            Freq_lbl = paste0(round(Freq*100), "%"),

            # Capitalize Gender variable
            Gender = str_to_title(Gender))  %>%

  #Finally, we can pipe it into ggplot
  #This method is equal to ggplot(data = df from the steps above, it saves you a bit of time and doesn'

ggplot() +
  geom_col(aes(x = Gender, y = Freq, fill = Gender), position = "dodge", color= "black") +
  # add labels, make a vertical adjustment
  geom_text(aes(x = Gender, y = Freq, label = Freq_lbl), vjust = -1) +
  labs(x = "\n% Who do smoke",
       y = "Proportion",
       title = "Faceted Bar Plot of Smoking Status by Gender using facet_wrap with proportions") +

  # use percentages for the y-axis labels, have the axis run from 0 to 70%
  scale_y_continuous(labels = scales::percent, limits = c(0, 0.7)) +

  scale_fill_manual(values = c("Male" = "purple", "Female" = "orange")) +
  facet_wrap(~ Race3, ncol = 3) + # 3 columns
  theme_classic() +
  theme(legend.position = "bottom") +

  # don't show legend for fill since it would be redundant
  guides(fill = "none")
```

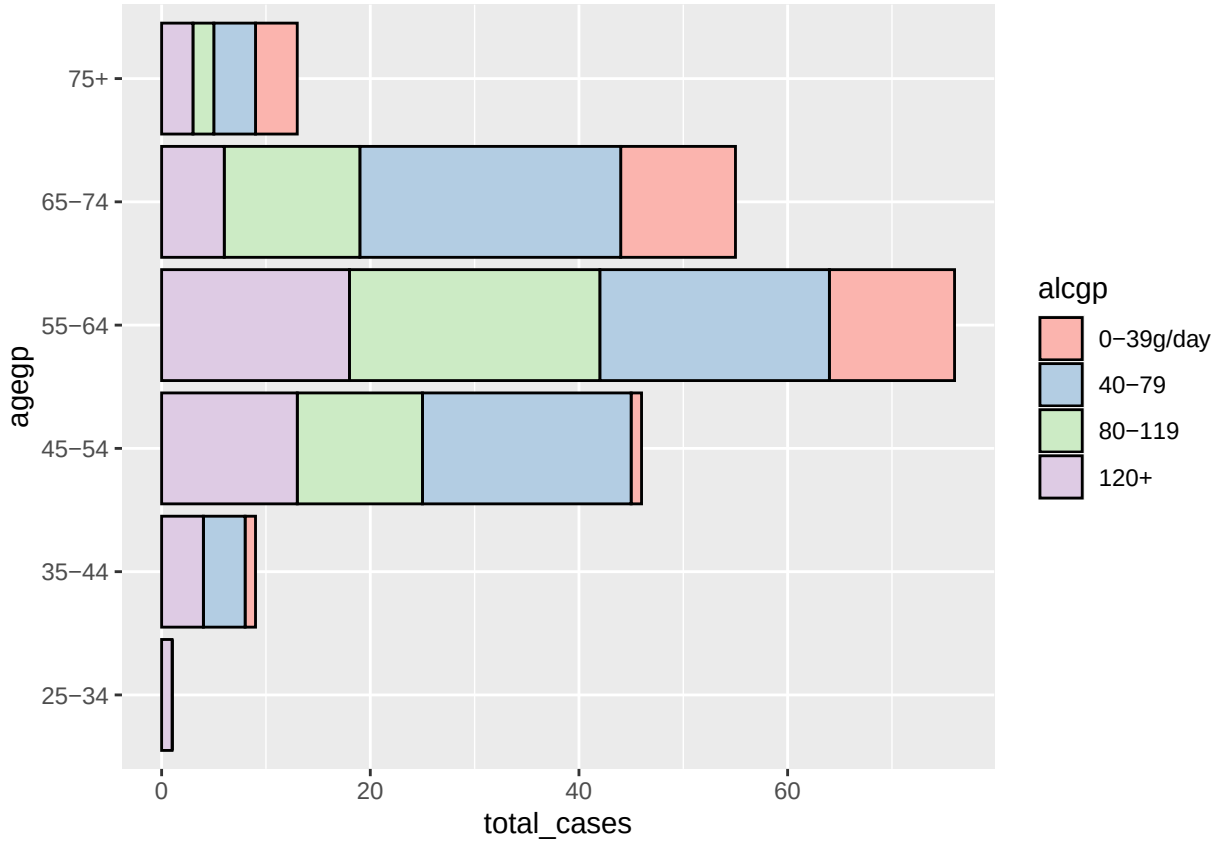
```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call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

## `summarise()` has grouped output by 'Gender', 'Race3'. You can override using
## the `.groups` argument.
```



Note that this changes our interpretation quite a bit – it seems that females across different race/ethnicity groups all tend to proportionally smoke less than their male counterparts.

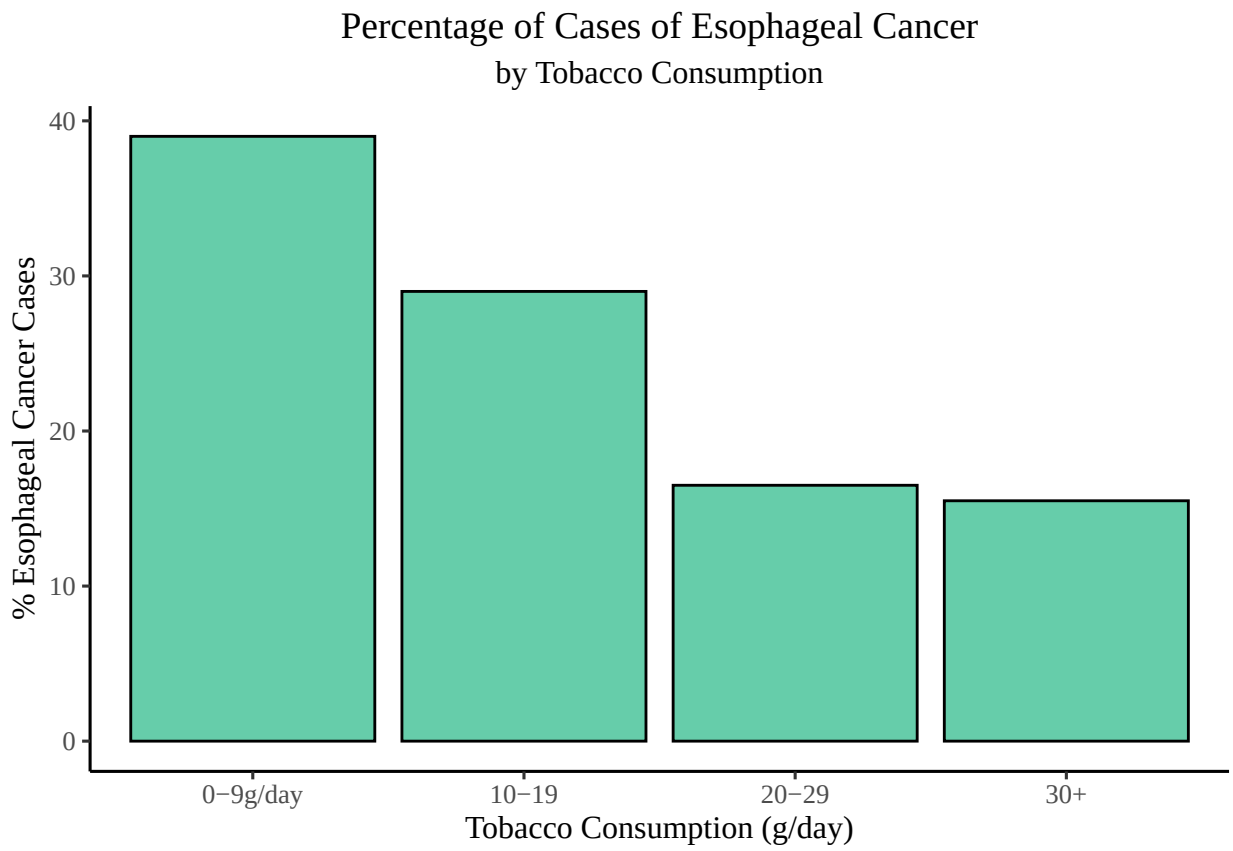
3.4: Horizontal bar chart

```
# Flipping the axis of the barchart
barchart3 +
  coord_flip() # Mention this to swap the x and y axes
```


75+
65−74
55−64
45−54
35−44
25−34
agegp
0
20
40
60
total_cases
alcgp
0−39g/day
40−79
80−119
120+

Module Summary/Recap

- We have reviewed how to customize `ggplot` figures for publications or presentation to a professional audience
- We explored modifications to color, labels, appearance, and legends
- You should be able to produce a polished bar chart that is publication-worthy (run the code below for a professional bar chart)



Assignment: Module Questions

Question 1: Create Bar Chart

TRY-IT-YOURSELF: We have individually gone through the methods to customize and elevate our plots. So, to produce a final, professional bar chart, can you combine all of the modifications (i.e. Captions/Labels, Color, and Legend) above into a single call?

```
# Type your answer here
```

Question 2: Create Stacked Bar Chart

TRY-IT-YOURSELF: Now that we have reviewed stacked barcharts, it's time for you to create and modify one! Produce a professional stacked bar chart that shows the number of cases of esophageal cancer by each age group's tobacco consumption.

```
# Type your answer here
```

Question 3: Create Dodged Bar Chart

TRY-IT-YOURSELF: You have already created a stacked bar chart for tobacco consumption. Similarly, now that we have gone through grouped bar charts, produce a professional grouped barchart that shows the number of cases of esophageal cancer by each age group's tobacco consumption.

```
# Type your answer here
```