Bivariate Categorical data (Part 2 of 2)

Chapter 9.

Exploring Multivariate Categorical Data

This chapter explores how to summarize and visualize *multivariate, categorical* data.

1. Multivariate categorical variables allow us to analyze relationships between **three or more** categorical variables respectively.

2. This form of analysis can help **reveal complex interactions and dependencies** between multiple variables that cannot be detected in bivariate analyses. Both bivariate and multivariate analyses are essential in statistical and data analysis as they allow us to uncover relationships and patterns in data. [1]

3. **Data**: Suppose we run the following code to prepare the `mtcars` data for subsequent analysis and save it in a tibble called `tb`. [2]

```
# Load the required libraries, suppressing annoying startup messages
library(dplyr, quietly = TRUE, warn.conflicts = FALSE)
library(tibble, quietly = TRUE, warn.conflicts = FALSE)
# Read the mtcars dataset into a tibble called tb
data(mtcars)
tb <- as_tibble(mtcars)

# Convert relevant columns into factor variables
tb$cyl <- as.factor(tb$cyl) # cyl = {4,6,8}, number of cylinders
tb$am <- as.factor(tb$am) # am = {0,1}, 0:automatic, 1: manual transmission
tb$vs <- as.factor(tb$vs) # vs = {0,1}, v-shaped engine, 0:no, 1:yes
tb$gear <- as.factor(tb$gear) # gear = {3,4,5}, number of gears

# Directly access the data columns of tb, without tb$mpg
attach(tb)
```

Three Way Relationships

1. A Three-Dimensional Contingency Table, often referred to as a **3-way contingency table**, is a statistical tool that helps analyze the relationship between three categorical variables. It builds upon the concept of a standard two-dimensional contingency table, which shows the distribution of two categorical variables, by adding a third dimension to the analysis.

2. Imagine a grid-like structure with three axes representing the three variables. The **rows** correspond to the categories of the first variable, the **columns** represent the categories of the second variable, and the **layers** (sheets) represent the categories of the third variable. Each cell within the table contains the frequency or count of observations that belong to a specific combination of the three variables.

3. When dealing with a three-way relationship, our focus is on three categorical variables and how they **interact** with each other. Such interactions can be manifest in the form of changes in the relationship between two variables based on the values of the third variable. Alternatively, we might seek to comprehend how all three variables collectively influence the observed data.

4. Graphically, three-way relationships can be represented in various forms, such as **three-dimensional contingency tables**, **side-by-side mosaic plots**, or even **three-dimensional bar plots**. However, it's crucial to note that these visual representations can become complicated and challenging to decipher as the number of categories within each variable rises. [1].

Three-Dimensional Contingency Tables

1. The R language, versatile as it is, provides multiple functions for creating contingency tables for multivariate categorical data. In this case, we're focusing on the `table()`, `xtabs()`, and `ftable()` functions for forming a three-way contingency table. [2]

2. We can create a three-way contingency table of `cyl`, `gear`, and `am` using the `table()` function.

```
# Create a contingency table showing the frequencies of car counts
# grouped by cylinder number ('cyl'), number of gears ('gear'),
# and transmission type ('am').
table(cyl,
      gear,
      am)
```

```
, , am = 0

    gear
cyl  3  4  5
  4  1  2  0
  6  2  2  0
  8 12  0  0

, , am = 1

    gear
cyl  3  4  5
  4  0  6  2
  6  0  2  1
  8  0  0  2
```

- When we run this code, the output is a **three-dimensional contingency table** showing the frequencies of all combinations of the three variables. Each cell in the resulting table represents the number of observations for a particular combination of `cyl`, `gear`, and `am` categories.

- Notice that we are segmenting the tables based on the 3rd argument given the table function, which is the transmission `am`.

3. We could alternately run the following code and instead segment the tables based on the cylinders `cyl`. [2]

```
# Create a contingency table with counts of cars grouped by
# transmission ('am'), number of gears ('gear'), and cylinder ('cyl')
table(am,
      gear,
      cyl)
```

```
, , cyl = 4

   gear
am   3  4  5
  0  1  2  0
  1  0  6  2

, , cyl = 6
```

```
      gear
am    3  4  5
  0   2  2  0
  1   0  2  1

, , cyl = 8

      gear
am    3  4  5
  0  12  0  0
  1   0  0  2
```

4. **xtabs()**: We can also create a three-way contingency table of `cyl`, `gear`, and `am` using the `xtabs()` function. [2]

```
# Use xtabs to create a cross-tabulation of cylinder count ('cyl'),
# gear count ('gear'), and transmission type ('am')
xtabs(~ cyl + gear + am,
      data = tb)
```

```
, , am = 0

     gear
cyl   3  4  5
  4   1  2  0
  6   2  2  0
  8  12  0  0

, , am = 1

     gear
cyl   3  4  5
  4   0  6  2
  6   0  2  1
  8   0  0  2
```

- Here, the formula `~ cyl + gear + am` defines the three variables we are interested in.

5. **ftable()**: The `ftable()` function is employed to generate a 'flat' contingency table, which is essentially a higher-dimensional contingency table displayed in a two-dimensional format. We can also create a three-way contingency table of `gear`, `cyl`, and `am` using the following code:

```
# Create a flat contingency table using 'ftable' to display the relationship between
# gear count ('gear'), cylinder count ('cyl'), and transmission type ('am')
ftable(gear + cyl ~ am,
        data = tb)
```

```
    gear  3           4           5
    cyl   4  6  8   4  6  8   4  6  8
am
0         1  2 12   2  2  0   0  0  0
1         0  0  0   6  2  0   2  1  2
```

- In this code, `ftable(gear + cyl \~ am, data = tb)`, we arrange the `gear` and `cyl` variables in the rows and the `am` variable in the columns.

- The `~` operator separates the variables that will be displayed in rows (on the left) and columns (on the right) in the resulting table.

- The `+` operator denotes that both `gear` and `cyl` will be included in the row labels. [2]

```
# Generate a flat contingency table displaying the relationship between
# gear ('gear') and a combination of cylinder ('cyl') and transmission ('am')
ftable(gear ~ cyl + am,
        data = tb)
```

```
        gear  3  4  5
cyl am
4   0         1  2  0
    1         0  6  2
6   0         2  2  0
    1         0  2  1
8   0        12  0  0
    1         0  0  2
```

- This variation of code, `ftable(gear ~ cyl + am, data = tb)`, it is structured slightly differently. Here, the `gear` variable forms the row and both `cyl` and `am` variables form the columns of the flat contingency table.

- In both scenarios, a `ftable` provides a more compact view of the three-way relationship among the `gear`, `cyl`, and `am` variables. However, the orientation of the variables in the rows and columns changes, providing different views of the data and potentially making certain patterns more evident depending on the question we're trying to answer.

- The exact choice between `ftable(gear + cyl ~ am, data = tb)` and `ftable(gear ~ cyl + am, data = tb)` will depend on what specific relationships we are most interested in exploring in our data. [2]

Visualization using a Faceted Bar Plot in `ggplot`

1. A Three-Dimensional Bar Plot is a generalization of a conventional two-dimensional bar graph, expanded into a third dimension. Rather than using bars at specific x coordinates in a two-dimensional plane, we utilize a grid of bars on the x-y plane, extending upwards in the z direction to indicate the data's magnitude. [4]

2. Here is some sample code:

```
# Load necessary package
library(ggplot2)
```

```
Attaching package: 'ggplot2'
```

```
The following object is masked from 'tb':

    mpg
```

```
# Create a table with count of each combination
count_df <- table(tb$cyl,
                   tb$am,
                   tb$gear)
# Convert table to a data frame for plotting
count_df <- as.data.frame.table(count_df)

# Rename columns for clarity
names(count_df) <- c("cyl", "am", "gear", "count")

# Create the plot using ggplot
ggplot(count_df,
       aes(x=cyl,
           y=count,
           fill=gear)) +
  geom_bar(stat="identity",
           position="dodge") +  # Use dodge position for side-by-side bars
  facet_grid(~am) +  # Create a facet for each transmission type
```
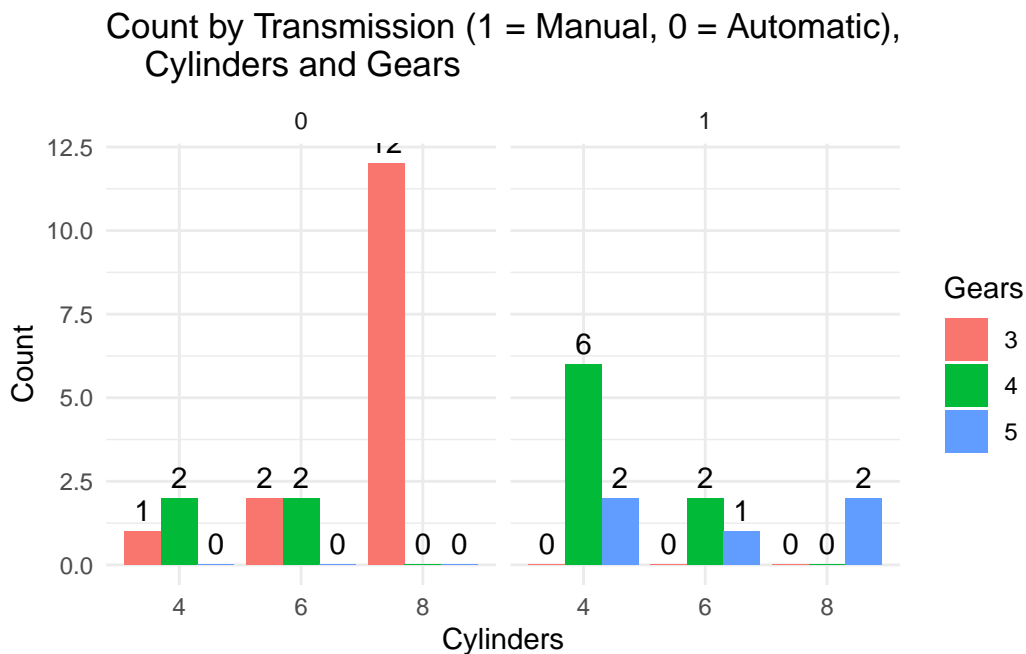
6

```r
  labs(
    title = "Count by Transmission (1 = Manual, 0 = Automatic),
    Cylinders and Gears",
    x = "Cylinders",
    y = "Count",
    fill = "Gears"
  ) +
# Add count labels to the bars
geom_text(aes(label = count),
          position = position_dodge(width = 0.9),
          vjust = -0.5) +  # Vertically adjust text to sit above bars
theme_minimal()  # Use a minimal theme for a clean look
```


Count by Transmission (1 = Manual, 0 = Automatic), Cylinders and Gears

- This code creates a faceted bar plot to visually represent the frequency of combinations of three categorical variables: `cyl` (Cylinders), `am` (Transmission), and `gear` (Gears).

- The line `count_df <- table(tb$cyl, tb$am, tb$gear)` generates a contingency table of the frequencies at each level of the three categorical variables (`cyl`, `am`, `gear`), using the `table()` function.

- Next, `count_df <- as.data.frame.table(count_df)` is used to convert the generated contingency table into a data frame, which can be more conveniently manipulated and visualized using `ggplot2`. [7]

- The names of the data frame's columns are then reassigned using `names(count_df) <- c("cyl", "am", "gear", "count")`. The 'count' column represents the frequency of each combination of the levels of the `cyl`, `am`, and `gear` variables.

- The plot is created using the `ggplot()` function, which initializes a ggplot object. The aesthetic mapping `aes(x=cyl, y=count, fill=gear)` specifies that the x-axis represents `cyl`, the y-axis represents `count`, and the color fill of the bars is based on `gear`.

- The `geom_bar(stat="identity", position="dodge")` function call adds a layer to the plot that depicts the data as a bar chart. The argument `stat="identity"` informs ggplot that the heights of the bars are given in the data (i.e., in the `count` variable), and `position="dodge"` causes bars associated with different levels of `gear` to be drawn side-by-side.

- The `facet_grid(~am)` function call adds facets to the plot based on the `am` variable, creating a separate subplot for each level of `am`.

- The `labs()` function call specifies the labels for the plot, including the title and the x-, y-, and fill-axis labels. The `theme_minimal()` call is used to apply a minimalist aesthetic theme to the plot.

- The `aes(label = count)` inside `geom_text()` tells ggplot to use the `count` column in the data frame as the labels for each bar. The `position_dodge(width = 0.9)` argument ensures that the labels are properly aligned with the corresponding bars in the grouped plot, and `vjust = -0.5` adjusts the vertical position of the labels to make them appear just above the bars.

- This code thus provides a clear and insightful visualization of the frequency of each combination of `cyl`, `am`, and `gear`. [4]

3. Here is some alternate sample code:

```
# Load necessary package
library(ggplot2)

# Create a table with count of each combination
count_df <- table(tb$cyl, tb$am, tb$gear)

# Convert table to a data frame for plotting
count_df <- as.data.frame.table(count_df)

# Rename columns for better understanding
names(count_df) <- c("cyl", "am", "gear", "count")

# Create the plot using ggplot2
```

```
ggplot(count_df, aes(x=am, y=count, fill=gear)) +
  geom_bar(stat="identity",
           position="dodge") +
  # Use dodge to display separate bars for each gear type
  facet_grid(~cyl) +
  # Facet by cylinder count for separate plots for each cylinder type
  labs(
    title = "Frequency by Cylinders, Transmission and Gears",
    x = "Tranmission",  # X-axis label
    y = "Count",  # Y-axis label
    fill = "Gears"  # Legend title
  ) +
  # Add count labels to each bar for clarity
  geom_text(aes(label = count),
            position = position_dodge(width = 0.9),
            vjust = -0.5) +  # Adjust the position of the text labels
  theme_minimal()  # Use a minimal theme for a clean appearance
```


Frequency by Cylinders, Transmission and Gears
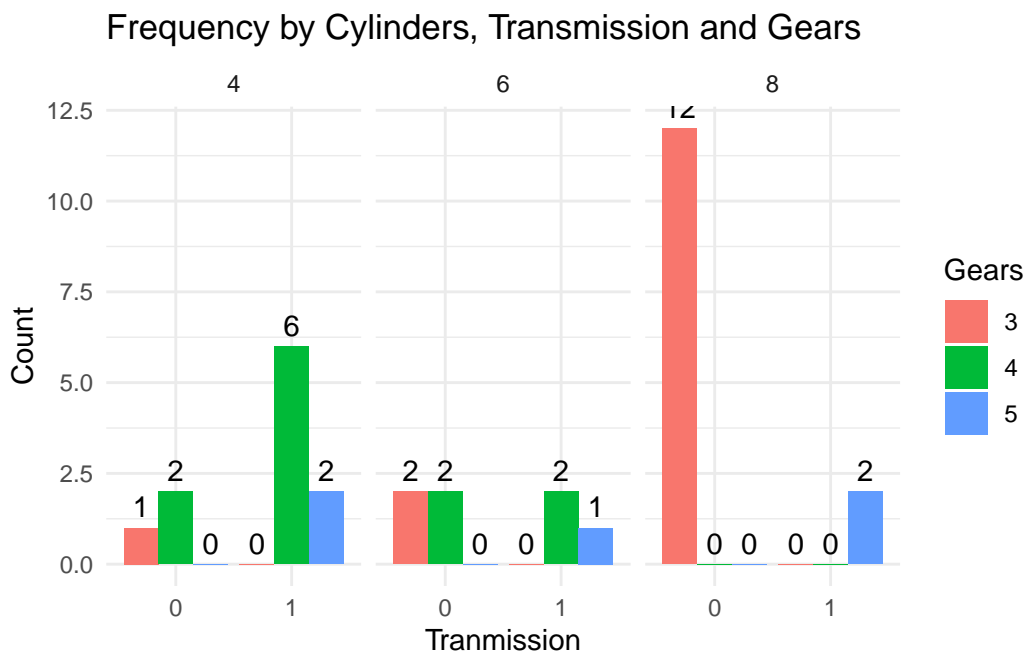
- This code is similar to the previous one; both create a faceted bar plot to display the frequencies of the levels of three categorical variables. The main difference between the two lies in the aesthetic mappings and the facet specification in the `ggplot()` function call.

- In this new code, the x-axis mapping in `aes()` is changed from `cyl` (Cylinders) to `am`

(Transmission). Hence, the x-axis of the bar plot will now depict the Transmission type instead of Cylinders.

- Similarly, the `facet_grid()` function, which was previously applied to `am`, is now applied to `cyl`. This means that the plot will now be faceted by the Cylinders variable. Each facet (or subplot) will correspond to a different number of Cylinders (4, 6, or 8). [4]

Visualization using a Mosaic Plot

```
# Load necessary packages
library(vcd)
```

Loading required package: grid

```
# Create a mosaic plot
vcd::mosaic(~gear+cyl+am,
        data = tb,
        main = "Mosaic Plot of 3 variables",  # Set the title
        shade = TRUE,  # Apply shading to highlight differences
        highlighting = "cyl" )  # Highlight based on cylinders
```
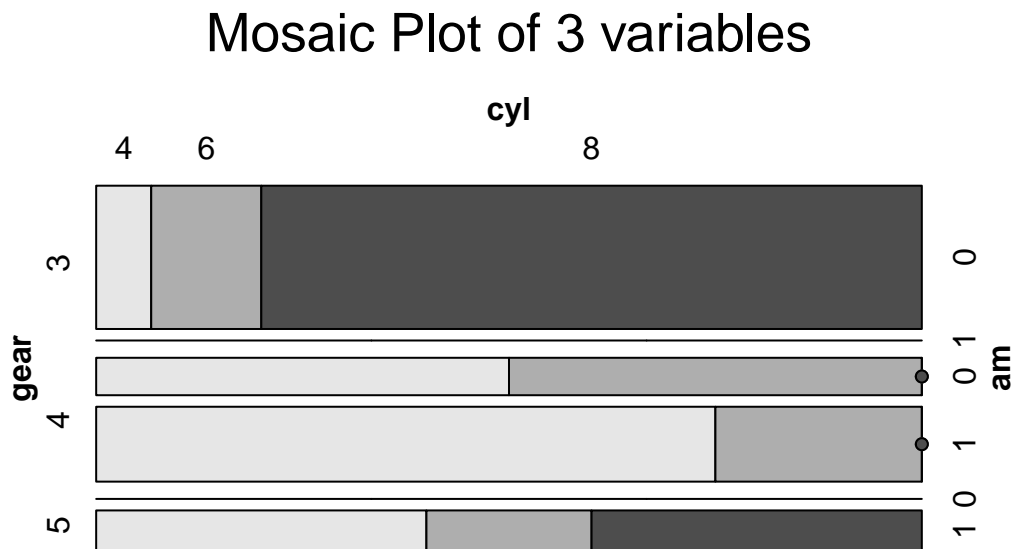

1. The provided R code generates a mosaic plot using the `vcd` package, specifically focusing on the `gear`, `cyl`, and `am` variables. A mosaic plot is a visual representation of the frequencies or proportions of combinations of categorical variables.

- `vcd::mosaic(~gear+cyl+am, data = tb, main = "Mosaic of Gears, Cylinders and Transmission type", shade = TRUE, highlighting = "cyl" )` generates the mosaic plot. The `~gear+cyl+am` formula indicates that the mosaic plot should visualize the `gear`, `cyl`, and `am` variables.

- `data = tb` specifies the dataset to be used, which is `tb` in this case.

- `main = "Mosaic of Gears, Cylinders and Transmission type"` sets the main title of the plot.

- `shade = TRUE` means that shading is applied to the cells in the plot. The shading can help to differentiate the cells visually based on the residuals from a model of independence.

- `highlighting = "cyl"` means that the `cyl` variable's levels will be distinctly colored. This highlighting helps to visually emphasize the differences among `cyl` categories in the plot. [5]

2. The generated mosaic plot provides an effective visual exploration of the joint distribution of `gear`, `cyl`, and `am` variables in the `tb` dataset, highlighting the `cyl` variable.

Four-Way Relationships

- Studying a four-way relationship between four categorical variables, involves looking at their interactions to see how they work together. [1]

Four-Dimensional Contingency Tables

1. Recall that `am`, `cyl`, `gear`, and `vs` are all categorical / factor variables.

- `am` indicates the type of transmission (0 = automatic, 1 = manual).
- `cyl` represents the number of cylinders in the car's engine.
- `gear` is the number of forward gears in the car.
- `vs` indicates the engine shape (0 = V-shaped, 1 = straight).

2. Suppose we wish to create a 4-way Contingency Table. Recall that the `ftable()` function provides an easy way to summarize categorical data in a "flat" table, which can make the data easier to understand and interpret. [2]

```
# Create a flat contingency table with transmission ('am') and
# cylinder ('cyl') against gear ('gear') and engine shape ('vs')
ftable(am + cyl ~ gear + vs,
       data = tb)
```

```
        am   0           1
        cyl  4  6  8   4  6  8
gear vs
3    0       0  0 12   0  0  0
     1       1  2  0   0  0  0
4    0       0  0  0   0  2  0
     1       2  2  0   6  0  0
5    0       0  0  0   1  1  2
     1       0  0  0   1  0  0
```

3. **Discussion**:

- In the formula `am + cyl ~ gear + vs`, the tilde (`~`) separates the rows and columns of the table.
- The variables to the left of the tilde (`am` and `cyl`) form the rows, and the variables to the right of the tilde (`gear` and `vs`) form the columns.
- The plus sign (`+`) between variables indicates that we want to consider all combinations of these variables.
- The resulting table shows the counts of each combination of `am` and `cyl` for each combination of gear and vs.

4. We can personalize the 4-way contingency table in several ways. Consider this alternate code. [2]

```
# Create a flat contingency table showing the relationship between
# transmission ('am'), cylinder ('cyl'), and engine shape ('vs')
# against the number of gears ('gear')
ftable(am + cyl + vs ~ gear,
       data = tb)
```

```
        am   0                         1
        cyl  4      6      8      4      6      8
        vs   0  1   0  1   0  1   0  1   0  1   0  1
gear
3            0  1   0  2  12  0   0  0   0  0   0  0
4            0  2   0  2   0  0   0  6   2  0   0  0
5            0  0   0  0   0  0   1  1   1  0   2  0
```

5. **Discussion**:

- This code `ftable(am + cyl + vs ~ gear, data = tb)` also uses the `ftable` function in R to generate a contingency table. However, there is a significant difference in the configuration of variables compared to the previous code.

- In the previous code, `am + cyl ~ gear + vs`, the row variables (`am` and `cyl`) were cross-tabulated against the combinations of column variables (`gear` and `vs`).

- In this new code, we have `am + cyl + vs ~ gear`, which means that we now have three row variables (`am`, `cyl`, and `vs`) and one column variable (`gear`). This will result in a table showing the counts of each combination of `am`, `cyl`, and `vs` for each level of `gear`.

- In short, the new code has added `vs` as a third row variable and removed it from the column variables, thus changing the structure and the interpretation of the resulting table. [3]

Visualization using a Mosaic Plot

1. We can visualize four dimensional Contingency Tables using a Mosaic Plot. This is an extension to our previous discussions about two-way and three-way contingency tables. [5]

Consider the following code:

```
# Create a mosaic plot of cyl vs, gear, am
vcd::mosaic(~ cyl + vs + gear + am,
            data = tb,
            main = "Mosaic Plot of 4 variables",  # Title
            shade = TRUE,  # Apply shading
            highlighting = "cyl" )  # Highlight based on cylinder
```

Mosaic Plot of 4 variables
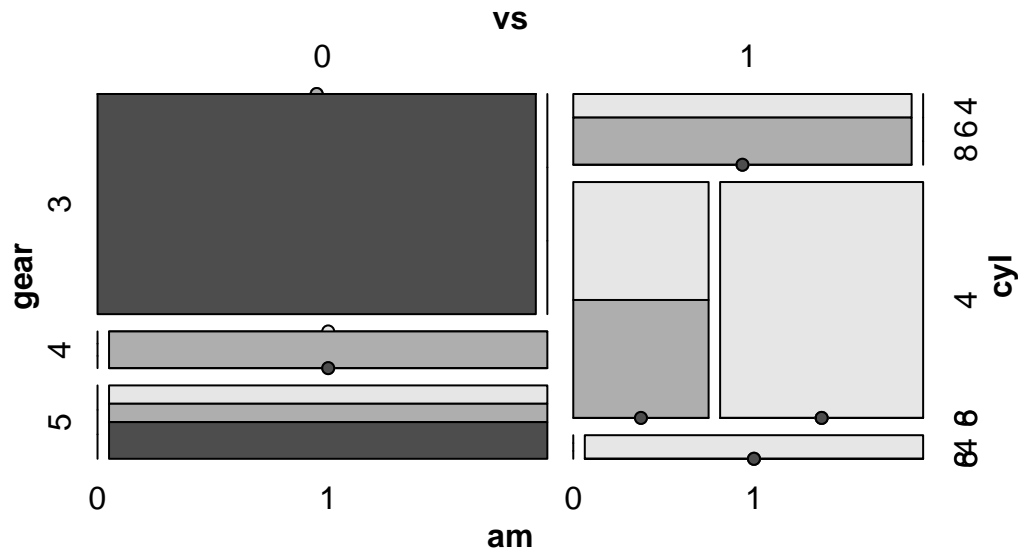
2. **Discussion**:

- `~ cyl + vs + gear + am` is a formula that indicates the categorical variables to be plotted. Each variable will be represented as a dimension in the plot, and their combined proportions will make up the whole plot.

- The `data = tb` argument informs the dataframe.

- The `main = "Mosaic Plot of 4 variables"` line sets the main title of the plot.

- `shade = TRUE` means that the cells in the mosaic plot will be shaded. The shading adds an extra visual element, which can make it easier to compare proportions.

- Finally, `highlighting = "cyl"` means that the cells in the plot will be highlighted based on the levels of the `cyl` variable. This can help to visually differentiate the categories of this variable. [5]

Summary of Chapter 9 – Bivariate Categorical data (Part 2 of 2)

In this chapter, we explore the world of multivariate categorical variables. Specifically, we focus on three-dimensional analyses that unveil complex relationships and dependencies between numerous variables. We utilize the R programming language, demonstrating a range of functions for constructing three-dimensional contingency tables. We also delve into segmenting these tables based on various variables, offering unique perspectives on the data.

Further, we expand on visualizing this data, discussing the creation of three-dimensional bar plots and mosaic plots. These powerful visual tools assist in data interpretation by representing the frequency of combinations of multiple variables. We don't limit ourselves to three-dimensional analysis; we advance into examining four-way relationships between categorical variables. Here, we utilize four-dimensional contingency tables and mosaic plots for analysis and visualization.

In essence, this chapter provides a robust understanding of multivariate categorical data handling and visualization, preparing readers to navigate and analyze complex datasets effectively. Overall, these three chapters together provide a comprehensive understanding of handling and visualizing multivariate categorical data.

References

Categorical Data Analysis:

[1] Agresti, A. (2018). An Introduction to Categorical Data Analysis (3rd ed.). Wiley.

Boston University School of Public Health. (2016). Analysis of Categorical Data. Retrieved from https://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/R/R6_CategoricalDataAnalysis/R6_CategoricalDataAnalysis2.html

Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2018). Multivariate Data Analysis (8th ed.). Cengage Learning.

Sheskin, D. J. (2011). Handbook of Parametric and Nonparametric Statistical Procedures (5th ed.). Chapman and Hall/CRC.

Tang, W., Tang, Y., & Song, X. (2012). Applied Categorical and Count Data Analysis. Chapman and Hall/CRC.

R Programming:

[2] Crawley, M. J. (2007). The R Book. Wiley.

Friendly, M., & Meyer, D. (2016). Discrete Data Analysis with R: Visualization and Modeling Techniques for Categorical and Count Data. Chapman and Hall/CRC.

Kabacoff, R. I. (2015). R in Action: Data Analysis and Graphics with R (2nd ed.). Manning Publications.

R Core Team. (2020). ftable: Flat Contingency Tables. In R Documentation. Retrieved from https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/ftable

Data Visualization:

[3] Chang, W. (2018). R Graphics Cookbook: Practical Recipes for Visualizing Data. O'Reilly Media.

Friendly, M. (2000). Visualizing Categorical Data. SAS Institute.

Healy, K., & Lenard, M. T. (2014). A Practical Guide to Creating Better Looking Plots in R. University of Oregon. Retrieved from https://escholarship.org/uc/item/07m6r

Healy, K. (2018). Data Visualization: A Practical Introduction. Princeton University Press.

Heiberger, R. M., & Robbins, N. B. (2014). Design of Diverging Stacked Bar Charts for Likert Scales and Other Applications. Journal of Statistical Software, 57(5), 1-32. doi: 10.18637/jss.v057.i05.

Lander, J. P. (2019). R for Everyone: Advanced Analytics and Graphics (2nd ed.). Addison-Wesley Data & Analytics Series.

Unwin, A. (2015). Graphical Data Analysis with R. CRC Press.

Wilke, C. O. (2019). Fundamentals of Data Visualization. O'Reilly Media.

ggplot2:

[4] Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4. Retrieved from https://ggplot2.tidyverse.org.

Wickham, H., & Grolemund, G. (2016). R for Data Science: Import, Tidy, Transform, Visualize, and Model Data. O'Reilly Media.

Wilkinson, L. (2005). The Grammar of Graphics (2nd ed.). Springer-Verlag.

Mosaic Plot:

[5] Hartigan, J. A., & Kleiner, B. (1981). Mosaics for Contingency Tables. In Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface (pp. 268-273).

Meyer, D., Zeileis, A., & Hornik, K. (2020). vcd: Visualizing Categorical Data. R Package Version 1.4-8. Retrieved from https://CRAN.R-project.org/package=vcd

Friendly, M. (1994). Mosaic Displays for Multi-Way Contingency Tables. Journal of the American Statistical Association, 89(425), 190-200.