Categorical Data (1 of 2)

July 26, 2023 V1.3 (Work in progress)

Overview

1. Categorical data is a type of data that can be divided into categories or groups. labels or categorical codes like "male" and "female," "red," "green," and "blue," or "A," "B," and "C" are frequently used to describe category data. This data type is commonly employed in research and statistics where you can categorize the data under various headings depending on their features, characteristics, or any other parameters [1]
2. There are several typical examples of categorical data.:
    - Gender (male, female)
    - Marital status (married, single, divorced)
    - Education level (high school, college, graduate school)
    - Occupation (teacher, doctor, engineer)
    - Hair color (brown, blonde, red, black)
    - Eye color (brown, blue, green, hazel)
    - Type of car (sedan, SUV, truck) [2]

Types of Categorical Data – Nominal, Ordinal Data

1. '**Nominal data**': This data type is characterized by variables that have two or more categories without having any kind of order or priority. The "nominal" term comes from the Latin "nomen," meaning "name." This name-based categorization means that nominal data represents whether a variable belongs to a certain category or not, but it does not convey any qualitative value about the variable itself. For example, 'gender' is a nominal data type, as it categorizes data into 'male' or 'female', and neither category is intrinsically superior to the other (Agresti, 2018). Nominal data is usually represented by text labels or categorical codes.

2. '**Ordinal data**': Unlike nominal data, ordinal data categories have an inherent order. This order, however, does not provide any information about the exact differences between the categories. Common examples of ordinal data include the Likert scale in surveys, which ranges from "very dissatisfied" to "very satisfied," or clothing sizes, which are typically "small," "medium," or "large." While we know that "large" is bigger than "medium" and "medium" is bigger than "small," we don't know exactly how much bigger one is compared to the other (McCrum-Gardner, 2008).

    - Ordinal data is frequently utilised to describe groups or categories that can be rated or sorted, such as educational level (high school, college, graduate school), or movie reviews (G, PG, PG-13, R, NC-17).

It's also worth noting that categorical data can be dichotomous (or binary), meaning there are only two possible categories. Both nominal and ordinal data can fall into this sub-type. A simple example would be a 'yes or no' survey response. [2]

Factor Variables in R

Factor variables in R programming language are essentially categorical variables. They are used to store categorical data, that is, data that falls into a limited number of categories or distinct groups (Wickham & Grolemund, 2016).

A factor variable has levels, which are the distinct categories of the variable. For instance, if we have a factor variable named "color," the levels might be "red," "blue," and "green." Each level corresponds to a category in the categorical data.

Creation of a factor variable in R can be done as follows:

```r
color <- c("red", "blue", "green", "red", "green", "yellow")
color_factor <- factor(color)
```

In this example, color_factor is a factor variable with four levels - "red", "blue", "yellow" and "green".

```r
levels(color_factor)
```

```
[1] "blue"   "green"  "red"    "yellow"
```

Consider the mtcars dataset. The following code prepares a tibble of car attributes for analysis, converting several of the variables into categorical data types using the factor structure in R.

```r
# Load the required libraries, suppressing annoying startup messages
library(tibble)
suppressPackageStartupMessages(library(dplyr))
```

```r
# Read the mtcars dataset into a tibble called tb
data(mtcars)
tb <- as_tibble(mtcars)
attach(tb)
```

```r
# Convert several numeric columns into factor variables
tb$cyl <- as.factor(tb$cyl)
tb$vs <- as.factor(tb$vs)
tb$am <- as.factor(tb$am)
tb$gear <- as.factor(tb$gear)
```

The above code reads the mtcars data into a tibble named tb and transforms certain variables from the tb tibble into factors (categorical variables), using the as.factor() function. Specifically, the cyl (cylinders), vs (engine shape), am (transmission type), and gear (number of gears) variables are transformed into factors. This change is useful for subsequent analyses that need to recognize these variables as categorical data, rather than numerical data.

Analysis of a Univariate Factor Variable

In the mtcars dataset, cyl stands for the number of cylinders in the car's engine. By transforming cyl into a factor variable, we're recognizing it as a categorical variable, which means we're acknowledging that it consists of several distinct categories or levels (Wickham & Grolemund, 2016). Each category or level corresponds to a specific number of cylinders a car might have.

Notice that in order to convert cyl into a factor variable in R, we have used the following line of code:

```r
tb$cyl <- as.factor(tb$cyl)
```

This line modifies the cyl column in the mtcars data frame (or tibble), transforming it from a numerical variable into a factor variable.

As a univariate factor variable, cyl represents a single, categorical characteristic of each car in the mtcars dataset: the number of cylinders in the engine. In this context, "univariate" means that we're only considering one variable at a time, without reference to any other variables (Sheskin, 2011). In the case of cyl, the unique categories (or levels) would correspond to the different numbers of cylinders in the cars' engines.

```r
levels(tb$cyl)
```

```
[1] "4" "6" "8"
```

The levels() function retrieves the levels of a factor variable. When it is applied to tb$cyl in the given context, it provides the unique categories of the cyl factor variable from the tb tibble, which are “4”, “6”, and “8”. These values correspond to the distinct number of cylinders in the cars’ engines that are represented in the mtcars dataset (Wickham & Grolemund, 2016).

Summarizing a univariate factor variable in R

A frequency table is a simple way to summarize categorical (or factor) data. It represents the count (frequency) of each category (level) in the factor variable. Essentially, a frequency table gives us a snapshot of the data distribution by indicating how many data points fall into each category (Crawley, 2007).

In R, we can create a frequency table using the table() function. For example, to create a frequency table for the cyl variable in the tb tibble, we would use:

```r
table(tb$cyl)
```

```

 4  6  8 
11  7 14 
```

The output shows the number of cars that fall into each cyl category (4, 6, or 8 cylinders). As an alternative, we can use the summary() function to create a summary table for a univariate factor variable. The summary() function provides a quick overview of a variable, including the frequency of each level for factor variables.

```r
summary(tb$cyl)
```

```
 4  6  8 
11  7 14 
```

Just like with table(), the output of summary() for a factor variable will show the number of occurrences for each level.

Proportions Table for One Variable

- `prop.table`
- Unlike table(), which delivers the count, this function returns the proportions of each category.

```
p0 = prop.table(table(cyl))
p0
```

```
cyl
      4       6       8 
0.34375 0.21875 0.43750 
```

The prop.table() function is used in this example to determine the percentage of each category in the cyl variable of the mtcars dataset. The fraction of cars with 4, 6, and 8 cylinders, respectively, is represented in the resulting vector p0. For instance, the dataset contains cars with 4 cylinders in 34.375% of the cases.

Rounding

This function is used to set the width of decimal numbers

round()

```
r1 = round(p0*100,2)
r1
```

```
cyl
    4     6     8 
34.38 21.88 43.75 
```

In this example, we round the proportion tables p0 and p1 to two decimal places using the round() method. The proportion of each category or group of categories, rounded to two decimal places, is presented in the ensuing tables r1 and r2. For instance, 56.25% of automobiles have an automated transmission and 8 cylinders.

Visualization of Categorical Variable

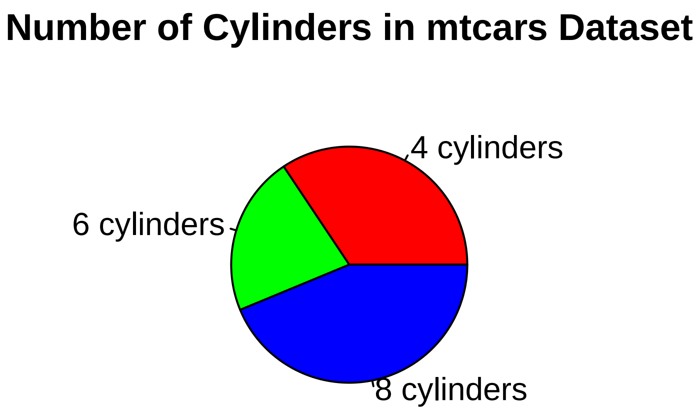
Pie chart

A pie chart is a circular graph with wedges or slices cut out of it, each of which represents a certain percentage of the entire. Each slice's size reflects the value it represents, while the chart's overall area reflects the sum of all values.

Pie charts are frequently used to display percentages or proportions of a whole or the relative sizes of various categories. They are very helpful for displaying data with few categories or when highlighting a single category or value.

```r
# Count the number of cars with each number of cylinders
cyl_counts <- table(mtcars$cyl)

# Create a pie chart
pie(cyl_counts, main = "Number of Cylinders in mtcars Dataset",
    labels = c("4 cylinders", "6 cylinders", "8 cylinders"),
    col = c("red", "green", "blue"))
```



The occurrences of each value of the cyl variable in the mtcars dataset are counted in this code using the table() function, and the resulting table is saved as cyl counts. The cyl counts variable provides the data for the pie chart, which is subsequently created using the pie() function. The chart's title is determined by the main argument, while the labels argument assigns unique labels to the chart's slices. The colours of the slices are specified by the col argument.

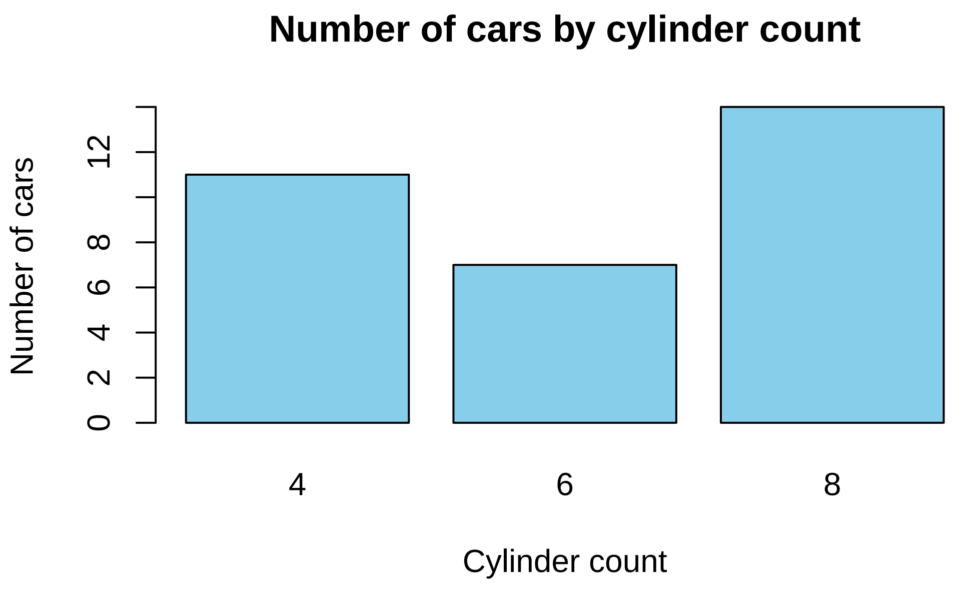
Barplot for categorical data in R

Barplot for Univariate Case

```
# Load the mtcars dataset
data(mtcars)

# Create a table of the counts of cars by number of cylinders
cyl_table <- table(mtcars$cyl)

# Create a barplot of the table
barplot(cyl_table,
        main = "Number of cars by cylinder count",
        xlab = "Cylinder count",
        ylab = "Number of cars",
        col = "skyblue")
```



By dividing the number of automobiles by the number of cylinders in the mtcars dataset, this code will produce a barplot. The barplot() function is used to generate the actual plot, while the table() function is used to generate a table of counts for the cyl variable in the mtcars dataset. The title and axis labels are added using the main, xlab, and ylab arguments, and the colour of the bars is altered with the col option.

References

[1] Diez, D. M., Barr, C. D., & Çetinkaya-Rundel, M. (2012). OpenIntro Statistics (2nd ed.). OpenIntro.

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

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

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

Healy, K., & Lenard, M. T. (2014). A practical guide to creating better looking plots in R. University of Oregon. https://escholarship.org/uc/item/07m6r

Few, S. (2004). Show me the numbers: Designing tables and graphs to enlighten. Analytics Press.

Friendly, M. (1994). Mosaic displays for multi-way contingency tables. Journal of the American Statistical Association, 89(425), 190-200.