STAT 3355
Introduction to Data Analysis

Lecture 06: Summaries for Univariate Data I

Created by: Qiwei Li
Assistant Professor of Statistics
Presented by: Octavious Smiley
Assistant Professor of Instruction

Department of Mathematical Sciences
The University of Texas at Dallas

Last Class

- Summarize a bivariate data

	Discrete + Discrete	Discrete + Continuous	Continuous + Continuous
Numerical			`cor(`x`, `y`)`
Graphical			`plot(`$y \sim x$`)` `abline((lm(`$y \sim x$`)))`

Learning Goals

- Numerical summaries for two discrete data
  - Contingency table
- Graphical summaries for two discrete data
  - Level plot
  - Stacked and side-by-side bar plots

Table of Contents

Discrete Data

- Samples share a finite number of values (have ties)

Discrete Data

- Samples share a finite number of values (have ties)
- Data type
  - Integer (if the number of unique values is small)
  - Categorical data
  - Logical data

Discrete Data

- Samples share a finite number of values (have ties)
- Data type
  - Integer (if the number of unique values is small)
  - Categorical data
  - Logical data
- Examples
  - The gender of a student: M or F
  - The blood type of a student: A, B, AB, or O
  - The STAT 3355 final grade of a student: A, B, C, D, E, or F
  - The political party that a student vote for: Democratic, republican, or independent

Discrete Data

- Samples share a finite number of values (have ties)
- Data type
  - Integer (if the number of unique values is small)
  - Categorical data
  - Logical data
- Examples
  - The gender of a student: M or F
  - The blood type of a student: A, B, AB, or O
  - The STAT 3355 final grade of a student: A, B, C, D, E, or F
  - The political party that a student vote for: Democratic, republican, or independent
- The distribution for a discrete data depends on the other discrete data

Data Tabulating

- Denote two univariate discrete dataset by

$$\boldsymbol{x} = [x_1, \ldots, x_i, \ldots, x_n], \text{ where } x_i \in \{0, 1, \ldots, K-1\}$$
$$\boldsymbol{y} = [y_1, \ldots, y_i, \ldots, y_n], \text{ where } y_i \in \{0, 1, \ldots, Q-1\}$$

Data Tabulating

- Denote two univariate discrete dataset by

$$\boldsymbol{x} = [x_1, \ldots, x_i, \ldots, x_n], \text{ where } x_i \in \{0, 1, \ldots, K-1\}$$
$$\boldsymbol{y} = [y_1, \ldots, y_i, \ldots, y_n], \text{ where } y_i \in \{0, 1, \ldots, Q-1\}$$

- Tabulating data is to obtain a contingency table (two-way table), which is essentially an integer matrix

$$\boldsymbol{F} = \begin{array}{c} \boldsymbol{x} \setminus \boldsymbol{y} \\ \text{Type } 0 \\ \text{Type } 1 \\ \vdots \\ \text{Type } K-1 \end{array} \begin{array}{c} \begin{array}{cccc} \text{Type } 0 & \text{Type } 1 & \cdots & \text{Type } Q-1 \end{array} \\ \begin{pmatrix} n_{0,0} & n_{0,1} & \cdots & n_{0,Q-1} \\ n_{1,0} & n_{1,1} & \cdots & n_{1,Q-1} \\ \vdots & \vdots & \ddots & \vdots \\ n_{K-1,0} & n_{K-1,1} & \cdots & n_{K-1,Q-1} \end{pmatrix} \end{array},$$

where $n_{k,q} = \sum_{i=1}^{n} I(x_i = k) I(y_i = q)$

Data Tabulating

- Denote two univariate discrete dataset by

$$\boldsymbol{x} = [x_1, \ldots, x_i, \ldots, x_n], \text{ where } x_i \in \{0, 1, \ldots, K-1\}$$
$$\boldsymbol{y} = [y_1, \ldots, y_i, \ldots, y_n], \text{ where } y_i \in \{0, 1, \ldots, Q-1\}$$

- Tabulating data is to obtain a contingency table (two-way table), which is essentially an integer matrix

$$\boldsymbol{F} = \begin{array}{c} \boldsymbol{x} \setminus \boldsymbol{y} \\ \text{Type } 0 \\ \text{Type } 1 \\ \vdots \\ \text{Type } K-1 \end{array} \begin{array}{c} \begin{array}{cccc} \text{Type } 0 & \text{Type } 1 & \cdots & \text{Type } Q-1 \end{array} \\ \begin{pmatrix} n_{0,0} & n_{0,1} & \cdots & n_{0,Q-1} \\ n_{1,0} & n_{1,1} & \cdots & n_{1,Q-1} \\ \vdots & \vdots & \ddots & \vdots \\ n_{K-1,0} & n_{K-1,1} & \cdots & n_{K-1,Q-1} \end{pmatrix} \end{array},$$

where $n_{k,q} = \sum_{i=1}^{n} I(x_i = k) I(y_i = q)$

- Interpretation
  - The frequency table of each pair of unique values in $\boldsymbol{x}$ and $\boldsymbol{y}$

Data Tabulating

- Implementation in R
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical vectors
    - `table(`x`,` y`)`
    - `xtabs(`$\sim$ x `+` y`)`
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical variables in a data frame $\boldsymbol{D}$
    - `table(`D`$x_name,` D`$y_name)`
    - `xtabs(`$\sim$ `x_name + y_name, data =` D`)`

Data Tabulating

- Implementation in R
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical vectors
    - `table(`x`, `y`)`
    - `xtabs(`$\sim$ x `+` y`)`
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical variables in a data frame $\boldsymbol{D}$
    - `table(`D`$x_name, `D`$y_name)`
    - `xtabs(`$\sim$ `x_name + y_name, data =` D`)`
  - `NA` will be omitted in the default setting

Data Tabulating

- Examples

```
# Load data samhda
data("samhda")

# Clean data
samhda$gender[which(samhda$gender == 7)] <-
    NA
samhda$alcohol[which(samhda$alcohol == 9)]
    <- NA
samhda$gender <- factor(samhda$gender,
    labels = c("M", "F"))
samhda$alcohol <- as.logical(2 - samhda$
    alcohol)
```

Data Tabulating

- Examples

```
# Obtain contigency table
F <- table(samhda$gender, samhda$alcohol)

# Obtain the relative frequency table
P <- F / sum(F)
```

Your Turn

- Continue to work on the dataset `samhda` in the package `UsingR`, which contains data on health behavior of school-aged children
  - Apply the function `table()` to the alcohol and marijuana variables, respectively. Do you need to clean the data?
  - Apply the function `table()` to the alcohol and marijuana variables together and get the contingency table
  - Read the contingency table or its relative frequency version, what's information you can tell?

Your Turn

- Solutions

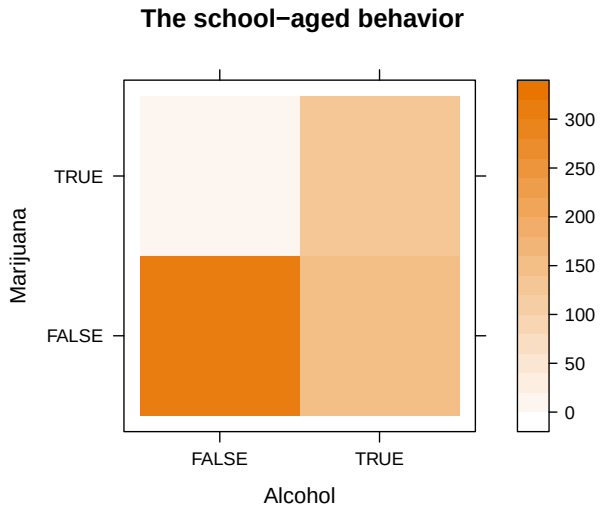
```
lean data
 samhda$alcohol[which(samhda$alcohol == 9)]
    <- NA
 samhda$alcohol[which(samhda$marijuana == 9)]
     <- NA
 samhda$alcohol <- as.logical(2 - samhda$
    alcohol)
 samhda$marijuana <- as.logical(2 - samhda$
    marijuana)

 # Obtain contigency table
 F <- table(samhda$alcohol, samhda$marijuana)
```

Level Plot

- Displays a surface in two dimensions

Level Plot

- Displays a surface in two dimensions
- Displays a numeric matrix in a grid
  - x axis arranges the levels of a discrete data in some order
  - y axis arranges the levels of the other discrete data in some order
  - The color in each grid indicates the corresponding entry's value



Level Plot

- Implementation in R
  - Install and load the package `lattice`
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical vectors
    - `levelplot(table(`x`,` y`))`
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical variables in a data frame $\boldsymbol{D}$
    - `levelplot(table(`D`$x_name,` D`$y_name))`

Level Plot

- Examples

```
# Load library
library(lattice)

# Obtain contigency table
F <- table(samhda$alcohol, samhda$marijuana)

# Plot the level plot
levelplot(F)
```

The Function `levelplot()`

- Important arguments controlling the levels
  - `cuts`: A integer that indicates the number of levels the range would be divided into
  - `col.regions`: A vector of gradually varying colors for the numbers
    - The number of entries should be greater than `cuts`; otherwise, the colors will be recycled
    - Create via `grey(level =)` and `level` is a vector of desired gray levels between 0 (black) and 1 (white)
    - Create via `colorRampPalette(colors =)` and `colors` is a vector of desired colors to interpolate
  - `alpha.regions`: A numeric number between 0 and 1 that specifies alpha transparency

Level Plot

- Examples

```
# Plot the level plot
levelplot(F)
levelplot(F, cuts = 3)
levelplot(F, col.regions = grey(level = seq(
    0, 1, by = 0.01)))
levelplot(F, col.regions = grey(level = rev(
    seq(0, 1, by = 0.01))))
levelplot(F, col.regions = colorRampPalette(
    colors = c("white", "red")))
levelplot(F, col.regions = colorRampPalette(
    colors = c("blue", "white", "red")))
levelplot(F, col.regions = colorRampPalette(
    colors = c("white", "#e87500")))
```

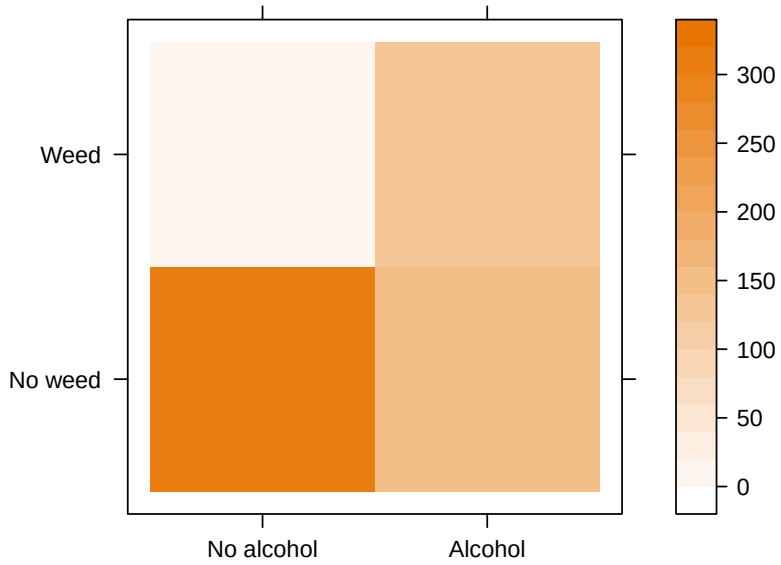
The Function `levelplot()`

- Important arguments controlling the labels
  - `main`: A string of title
  - `xlab` and `ylab`: A string of label for the axis names

Discussion

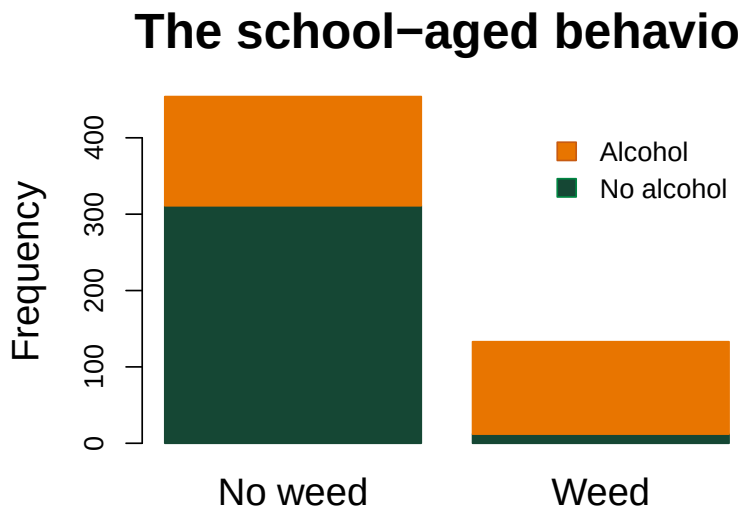
- The layout is different from the contingency table

	No weed	Weed
No al	311	12
Al	143	121



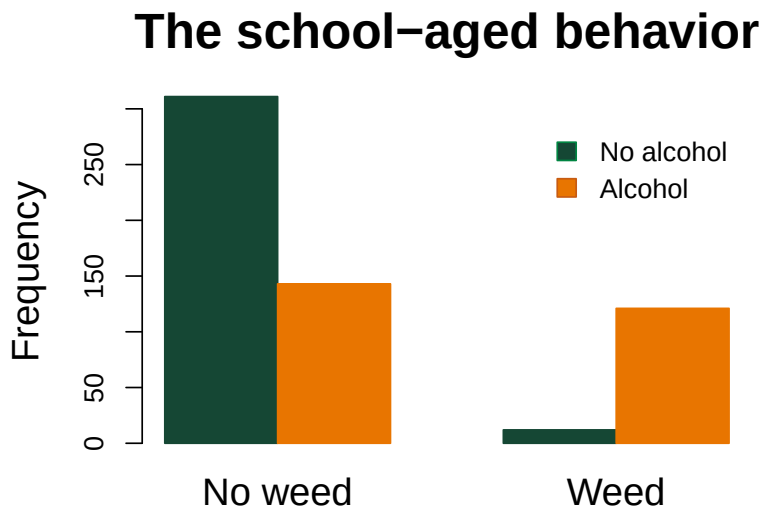
Bar Chart

- Stacked bar plot
  - One axis arranges the levels of the primary discrete data in some order
  - The other axis represents the frequency with a bar of a height proportional to the frequency
  - The colors indicate the levels of the secondary discrete data in some order



Bar Chart

- Side-by-side bar plot
  - One axis arranges the levels of the primary discrete data in some order, grouping the levels of the secondary discrete data
  - The other axis represents their frequency with a bar of a height proportional to the frequency
  - The colors indicate the levels of the secondary discrete data in some order



Bar Chart

- Implementation in R
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical vectors
    - `barplot(table(`x`, `y`), legend.txt = TRUE)`
  - $\boldsymbol{x}$ and $\boldsymbol{y}$ are integer/factor/logical variables in a data frame $\boldsymbol{D}$
    - `barplot(table(`D`$x_name, `D`$y_name), legend.txt = TRUE)`

Bar Chart

- Examples

```
# Obtain contigency table
F <- table(samhda$alcohol, samhda$marijuana)

# Plot the bar plot
barplot(F, names.arg = c("No Alcohol", "
    Alcohol"), legend.text = c("No weed", "
    Weed"))
barplot(F, beside = TRUE, names.arg = c("No
    Alcohol", "Alcohol"), legend.text = c("No
     weed", "Weed"))
```

The Function `barplot()`

- Important arguments controlling the bars
  - `horiz`: A logical value for the orientation of the bars
  - `beside`: A logical value for stacked or side-by-side plot
  - `col`: A vector of colors for each bar
  - `border`: A vector of colors for the boarder of each bar
  - `legend.text`: A logical value for displaying the legend or a character vector of names to construct the legend
  - `args.legend`: A list of additional arguments to pass to `legend`

The Function `barplot()`

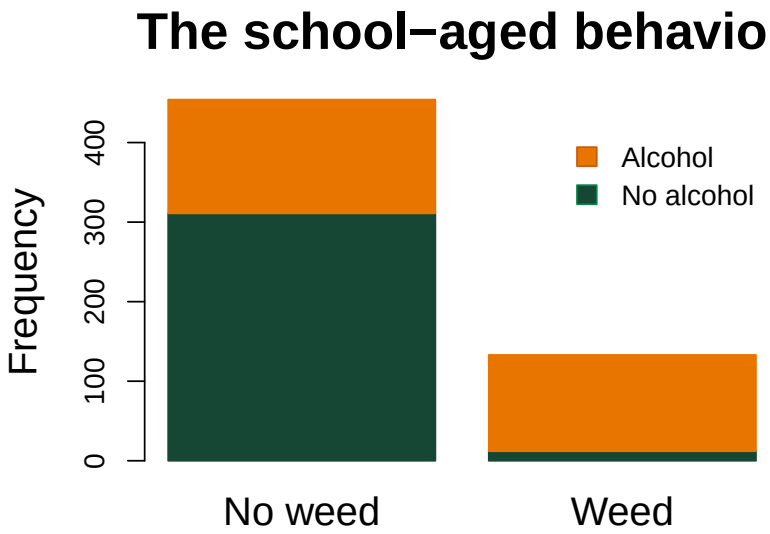
- Important arguments controlling the labels
  - `main`: A string of title
  - `names.arg`: A character vector of names for each bar
  - `xlab` and `ylab`: A string of label for the axis names
  - `las`: A numeric value of $\{0, 1, 2, 3\}$ for the orientation of axis labels

The Function `barplot()`

- Important arguments controlling the label size
  - `cex.main`: A numeric value for the title size
  - `cex.lab`: A numeric value for the size of axis labels
  - `cex.axis`: A numeric value for the size of x axis label
  - `cex.names`: A numeric value for the size of axis names

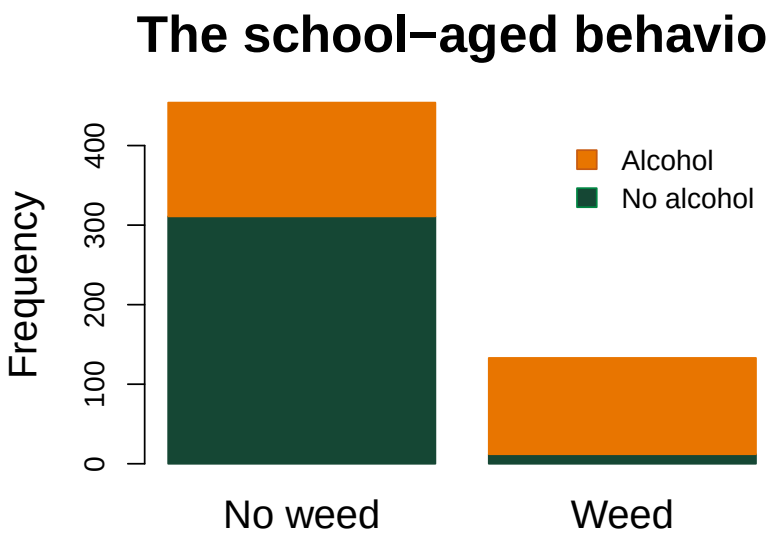
Discussions

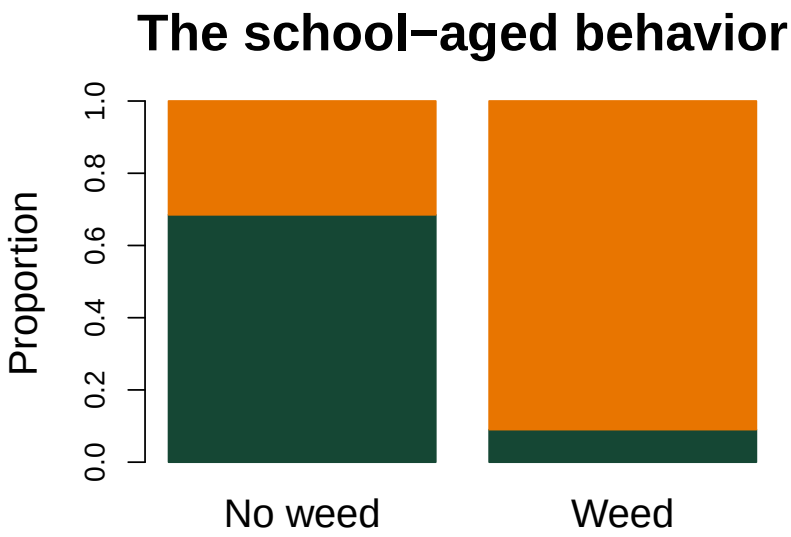
- Stacked bar plot is better for visualizing the conditional distributions



Discussions

- Stacked bar plot is better for visualizing the conditional distributions
- Consider to present the relative frequencies rather than frequencies via `prop.table(table(`x`, `y`), margin = 2)`





Discussions

- Examples

```
F <- table(samhda$alcohol, samhda$marijuana)
rownames(F) <- c("No alcohol", "Alcohol")
colnames(F) <- c("No weed", "Weed")

# Stacked bar plot
barplot(F, ylab = "Frequency", main = "The
    school-aged behavior", cex.names = 1.4,
    cex.axis = 1, cex.lab = 1.4, cex.main = 1
    .8, col = c("#154734", "#e87500"), border
     = c("#154734", "#e87500"), legend.text =
     TRUE, args.legend = list(border = c("#e8
    7500", "#154734"), bty = 'n'))
```

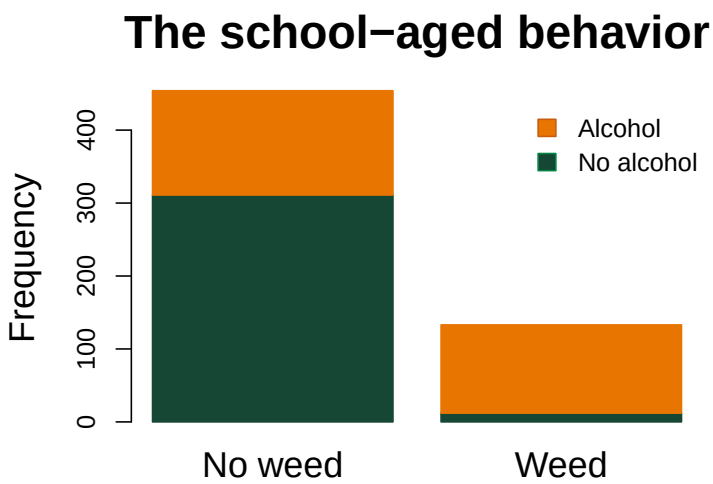
Discussions

- Examples

```
# Stacked barplot based on relative
    frequencies
barplot(prop.table(F, margin = 2), ylab = "
    Proportion", main = "The school-aged
    behavior", cex.names = 1.4, cex.axis = 1,
     cex.lab = 1.4, cex.main = 1.8, col = c("
    #154734", "#e87500"), border = c("#154734
    ", "#e87500"), legend.text = FALSE, args.
    legend = list(border = c("#e87500", "#154
    734"), bty = 'n'))
```

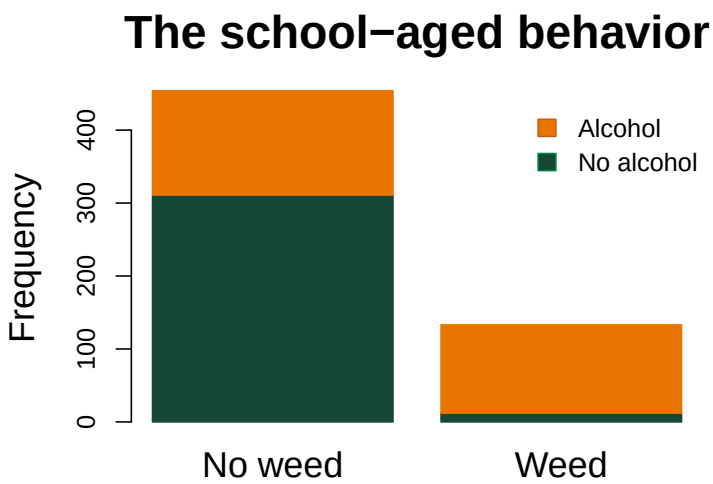
Discussions

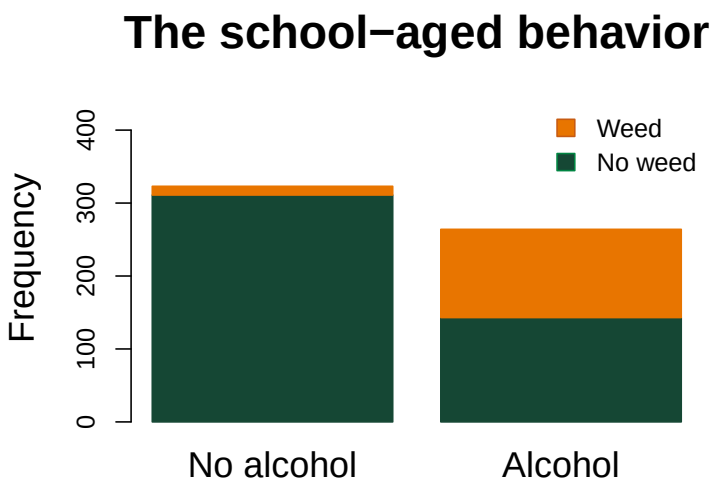
- Flip the contingency table to switch the primary variable
  - `table(`y`, `x`)`
  - `t(table(`x`, `y`))`



Discussions

- Flip the contingency table to switch the primary variable
  - `table(`y`, `x`)`
  - `t(table(`x`, `y`))`





Discussions

- Examples

```
# Stacked bar plot when the marijuana
    variable is the primary
barplot(F, ylim = c(0, 450), ylab = "
    Frequency", main = "The school-aged
    behavior", cex.names = 1.4, cex.axis = 1,
     cex.lab = 1.4, cex.main = 1.8, col = c("
    #154734", "#e87500"), border = c("#154734
    ", "#e87500"), legend.text = TRUE, args.
    legend = list(x = 2.5, y = 450, border =
    c("#e87500", "#154734"), bty = 'n'))
```

Discussions

- Examples

```
# Stacked bar plot when the alcohol variable
    is the primary
barplot(t(F), ylim = c(0, 450), ylab = "
   Frequency", main = "The school-aged
   behavior", cex.names = 1.4, cex.axis = 1,
    cex.lab = 1.4, cex.main = 1.8, col = c("
   #154734", "#e87500"), border = c("#154734
   ", "#e87500"), legend.text = TRUE, args.
   legend = list(x = 2.5, y = 450, border =
   c("#e87500", "#154734"), bty = 'n'))
```

After-class Reading

- *Using R for Introductory Statistics (1st Ed.)* by John Verzani
- Chapter 3 Bivariate data
  - Section 3.1 Pairs of categorical variables
    - Subsection 3.1.2 Making two-way tables from unsummarized data
    - Subsection 3.1.3 Marginal distributions of two-way tables
    - Subsection 3.1.4 Conditional distributions of two-way tables
    - Subsection 3.1.5 Graphical summaries of two-way contingency tables

After-class Reading

- *Using R for Introductory Statistics (2nd Ed.)* by John Verzani
- Chapter 3 Bivariate data
  - Section 3.4 Bivariate categorical data