Cookbook Data Analysis with Stata and R

Manuel Oliveira

2024-08-01

Table of contents

Welcome

Welcome to the "Cookbook Data Analysis with Stata and R"! This book is designed to be your comprehensive guide to mastering data analysis using two of the most powerful tools available: Stata and R. Whether you are a beginner or have some experience, this book will help you develop the skills needed to tackle a wide range of data analysis challenges.

Why This Book?

In the Human Technology Interaction track at TU/e, understanding and analyzing data is crucial. This book aims to provide you with practical, hands-on experience in data analysis, tailored specifically to the needs of your coursework and future career. By the end of this book, you will be able to:

- Import and manage data in both Stata and R.
- Clean and prepare your data for analysis.
- Perform a variety of statistical analyses.
- Visualize your data to uncover insights.
- Communicate your findings effectively.

What You Will Learn

Data analysis is a vast field, and this book focuses on giving you a solid foundation in the most essential tools and techniques. Here's what you can expect to learn:

1. **Data Import and Management**: Learn how to import data from various sources and manage it efficiently in Stata and R.
2. **Data Cleaning and Preparation**: Understand the importance of tidy data and how to clean and prepare your datasets for analysis.
3. **Statistical Analysis**: Perform descriptive and inferential statistics to draw meaningful conclusions from your data.
4. **Data Visualization**: Create compelling visualizations to explore and present your data.
5. **Reproducible Research**: Learn best practices for ensuring your analyses are reproducible and transparent.

Real-World Applications

Throughout this book, you will work on simulated dataset aiming to reflect real-world examples and case studies relevant to Human Technology Interaction. These examples will help you see how the techniques you learn can be applied to actual research and industry scenarios. Whether you are analyzing user behavior, evaluating the effectiveness of a new technology, or exploring human-computer interaction, this book will provide you with the tools you need.

Getting Started

To get the most out of this book, you will need to have Stata and R installed on your computer. We recommend using the latest versions of both software packages. Additionally, familiarity with basic statistical concepts will be helpful, but not required, as we will cover the necessary background along the way.

1 Introduction to R and Stata

2 Brief History of R and Stata

2.1 History of R

R is a programming language and free software environment for statistical computing and graphics. It was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and the first public announcement of R was made in 1993[1][3]. The language was inspired by the S language, which was developed at Bell Laboratories. R has since become a popular tool among statisticians and data scientists due to its extensive package ecosystem and active user community.

2.2 History of Stata

Stata is a general-purpose statistical software package created in the mid-1980s by William Gould, a UCLA graduate, and Sean Becketti[2][1]. Initially developed within the Computing Resource Center (CRC) in Santa Monica, California, Stata was designed to provide a powerful yet user-friendly environment for data analysis. The first version of Stata was released in 1985, and it has since evolved to include a wide range of statistical, graphical, and data management capabilities.

2.3 Comparison of R and Stata

Feature	R	Stata
Cost	Free and open-source	Paid software
User Interface	Command-line interface, with various IDEs like RStudio available	User-friendly GUI and command-line interface
Flexibility	Highly flexible, suitable for custom analyses and new methods	Less flexible, but highly efficient for standard statistical tasks
Packages	Extensive package ecosystem for various analyses and visualizations	Comprehensive built-in functions and user-written commands
Learning Curve	Steeper learning curve, especially for beginners	Easier to learn, with simpler syntax

Feature	R	Stata
Community Support	Large, active community with extensive online resources	Smaller community, but excellent official documentation
Reproducibility	Strong support for reproducible research through RMarkdown and other tools	Good support for reproducibility, but less integrated than R
Data Management	Powerful data manipulation capabilities with packages like dplyr	Efficient data manipulation and management built-in
Visualization	Advanced visualization capabilities with ggplot2 and other packages	Good visualization tools, but less flexible than R

[2][1]: A brief history of Stata on its 20th anniversary. The Stata Journal (2005). [1][3]: History and Overview of R. R Programming for Data Science - Bookdown.

3 Chapter 1: Getting Started

3.1 Introduction

This chapter provides a quick tutorial on how to install and set up R and Stata on both Windows and Mac computers. By the end of this chapter, you'll have the necessary tools ready to begin your analysis.

3.2 Installing R

3.2.1 Windows

1. **Download R**:
   - Go to the R Project website.
   - Click on "Download R for Windows."
   - Click on "base" to download the base R package.
2. **Install R**:
   - Run the downloaded `.exe` file.
   - Follow the installation instructions, accepting the default settings.
3. **Install RStudio** (Optional but recommended):
   - Download RStudio from the RStudio website.
   - Run the installer and follow the setup instructions.

3.2.2 Mac

1. **Download R**:
   - Visit the R Project website.
   - Click on "Download R for macOS."
2. **Install R**:
   - Open the downloaded `.pkg` file.

- Follow the installation instructions.

3. **Install RStudio** (Optional but recommended):
   - Download RStudio from the RStudio website.
   - Open the `.dmg` file and drag RStudio to your Applications folder.

3.3 Installing Stata

3.3.1 Windows

1. **Obtain a License**:
   - Stata is commercial software. Ensure you have a valid license.
2. **Download Stata**:
   - Go to the Stata website and log in to your account to download the installer.
3. **Install Stata**:
   - Run the downloaded `.exe` file.
   - Follow the installation instructions, entering your license information when prompted.

3.3.2 Mac

1. **Obtain a License**:
   - Make sure you have a valid license for Stata.
2. **Download Stata**:
   - Visit the Stata website and log in to your account to download the installer.
3. **Install Stata**:
   - Open the downloaded `.dmg` file.
   - Drag the Stata application to your Applications folder.
   - Launch Stata and enter your license information.

3.4 Setting Up Your Environment

3.4.1 R Setup

1. **Open RStudio** (or R GUI if not using RStudio).
2. **Install Essential Packages**:

    - Open the Console and run:

```r
install.packages(c("tidyverse", "lme4", "ggplot2"))
```

3. **Create a New Project** (Optional but recommended in RStudio):

    - Go to "File" > "New Project" > "New Directory" > "New Project."
    - Choose a location and name for your project, then click "Create Project."

3.4.2 Stata Setup

1. **Open Stata**.
2. **Set a Working Directory**:

    - Use the command:

```stata
cd "path/to/your/directory"
```

Replace `"path/to/your/directory"` with the path where you want to save your files.

3. **Creating Do-Files**:

    - Go to "File" > "New Do-file Editor."
    - Save the Do-file in your working directory.

3.5 Verification

3.5.1 R

1. **Test Installation**:

    - In RStudio or R GUI, type:

```r
print("R is working!")
```

- If you see the output `[1] "R is working!"`, your installation is successful.

2. **Load a Package**:
    - Run:

```
library(ggplot2)
print("ggplot2 is loaded!")
```

3.5.2 Stata

1. **Test Installation**:
    - In the Command window, type:

```
display "Stata is working!"
```

- If you see the output `Stata is working!`, your installation is successful.

2. **Check Version**:
    - Type:

```
about
```

- This will display the version of Stata installed.

With your environment set up, you're now ready to start performing analyses using R and Stata!

4 t-test

4.1 Brief Explanation

The t-test, proposed by William Sealy Gosset under the pseudonym "Student" in 1908, is used to determine if there is a significant difference between the means of two groups. It is commonly used when the sample sizes are small and the population variance is unknown.

Statistic	Description
Proposed by	William Sealy Gosset (1908)
Purpose	Compare means of two groups
When to use	Small sample sizes, unknown population variance
Example question	Is there a significant difference in user satisfaction between two versions of a software?
Analytical goal	Determine if the mean satisfaction scores differ significantly between the two versions

4.2 Research Question and Hypothesis

In today's fast-paced digital world, user experience (UX) is a critical factor that can make or break a product. Imagine a tech company, "InnovateTech," which has recently launched a new interface design for its flagship software. The company is keen to understand whether this new design truly enhances user satisfaction compared to the old design.

InnovateTech has invested significant resources into redesigning its software interface, aiming to make it more intuitive, visually appealing, and user-friendly. The old design, while functional, received mixed reviews from users, with common complaints about its complexity and outdated look. The new design promises a sleek, modern interface with improved navigation and enhanced features.

To validate the effectiveness of the new design, InnovateTech conducts a study involving a diverse group of users. Participants are randomly assigned to use either the old or the new interface for a week. At the end of the week, they complete a detailed satisfaction survey. The company hopes that the results will provide clear insights into whether the new design meets user expectations and enhances their overall experience.

Research Question: Does the new interface design improve user satisfaction compared to the old design?

Hypothesis: Users will report higher satisfaction scores with the new interface design compared to the old design.

4.3 R packages

```
# Load the necessary packages
library(tidyverse) # used for data manipulation and visualization
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.0     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(ggrain) # used for raincloud plots
```

```
Registered S3 methods overwritten by 'ggpp':
  method                  from
  heightDetails.titleGrob ggplot2
  widthDetails.titleGrob  ggplot2
```

```
library(cowplot) # for cowplot theme in ggplot
```

```

Attaching package: 'cowplot'

The following object is masked from 'package:lubridate':

    stamp
```

```
library(Statamarkdown) # to run Stata commands in an R environment
```

```
Stata found at C:/Program Files/Stata18/StataBE-64.exe
The 'stata' engine is ready to use.
```

```
# Statamarkdown configuration
stataexe <- "C:/Program Files/Stata18/StataBE-64.exe" # Add your own path to the Stata execu
knitr::opts_chunk$set(engine.path=list(stata=stataexe))

# to install any missing packages go to the Terminal and run the command: install.packages("
```

4.4 Simulated Dataset

4.4.1 In R

```
set.seed(123)
n <- 30
old_design <- rnorm(n, mean = 70, sd = 10)
new_design <- rnorm(n, mean = 75, sd = 10)
data <- data.frame(
  group = rep(c("Old Design", "New Design"), each = n),
  satisfaction = c(old_design, new_design)
)
write.csv(data, "satisfaction_data.csv", row.names = FALSE)
```

4.4.2 In Stata

```
clear
set seed 123
set obs 30
gen group = "Old Design"
gen satisfaction = rnormal(70, 10)
save old_design.dta, replace
```

```
Number of observations (_N) was 0, now 30.
```

```
file old_design.dta saved
```

```
clear
set obs 30
gen group = "New Design"
gen satisfaction = rnormal(75, 10)
save new_design.dta, replace
```

```
Number of observations (_N) was 0, now 30.
```

```
file new_design.dta saved
```

```
use old_design.dta
append using new_design.dta
save satisfaction_data.dta, replace
```

```
file satisfaction_data.dta saved
```

4.5 Statistical Analysis

4.5.1 Inspecting Data Descriptives and Plotting

4.5.1.1 In Stata

```
use satisfaction_data.dta
summarize satisfaction
graph box satisfaction, over(group)
```

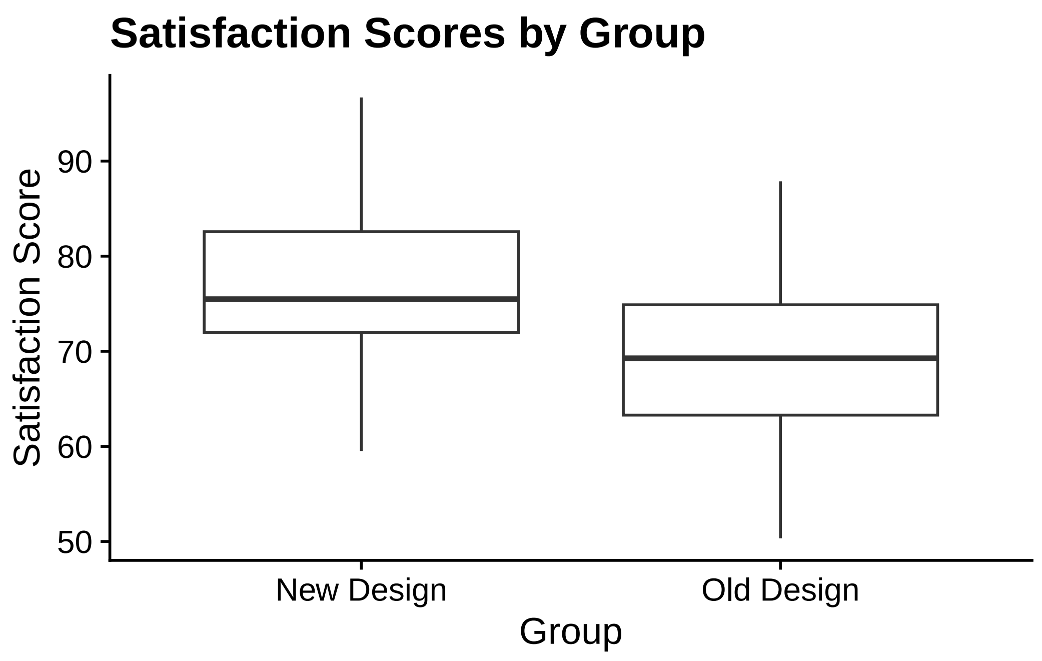
```
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
satisfaction |         60    72.69432    11.03765   50.47923   99.36316
```

4.5.1.2 In R

```r
data <- read.csv("satisfaction_data.csv")
summary(data$satisfaction)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  50.33   65.48   73.26   73.16   80.62   96.69 
```

```r
p <- ggplot(data, aes(x = group, y = satisfaction)) +
  geom_boxplot() +
  theme_cowplot() +
  labs(title = "Satisfaction Scores by Group", x = "Group", y = "Satisfaction Score")
p
```



4.5.2 T-test

4.5.2.1 In Stata

```
use satisfaction_data.dta
ttest satisfaction, by(group)
```

```
Two-sample t test with equal variances
------------------------------------------------------------------------------
   Group |     Obs        Mean    Std. err.   Std. dev.   [95% conf. interval]
---------+--------------------------------------------------------------------
New Desi |      30    75.08677    2.226942    12.19746    70.53216    79.64138
Old Desi |      30    70.30186    1.705283    9.340221    66.81417    73.78956
---------+--------------------------------------------------------------------
Combined |      60    72.69432    1.424954    11.03765    69.84299    75.54564
---------+--------------------------------------------------------------------
    diff |            4.784905    2.804864               -.8296399    10.39945
------------------------------------------------------------------------------
    diff = mean(New Desi) - mean(Old Desi)                        t =   1.7059
H0: diff = 0                                     Degrees of freedom =       58

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.9533         Pr(|T| > |t|) = 0.0934          Pr(T > t) = 0.0467
```

4.5.2.2 In R

```
t.test(satisfaction ~ group, data = data)
```

```

	Welch Two Sample t-test

data:  satisfaction by group
t = 3.0841, df = 56.559, p-value = 0.003156
alternative hypothesis: true difference in means between group New Design and group Old Desi
95 percent confidence interval:
  2.543416 11.965426
sample estimates:
mean in group New Design mean in group Old Design
                76.78338                 69.52896
```

4.6 Explanation of Relevant Terms

Term	Definition	Common Misconception
P-value	The probability of obtaining test results at least as extreme as the results actually observed, under the assumption that the null hypothesis is true.	The p-value is the probability that the null hypothesis is true.
Confidence Interval	A range of values, derived from the sample data, that is believed to contain the true parameter value with a certain probability. If repeated samples were taken, a specified proportion of these intervals would contain the true parameter value.	A 95% confidence interval means there is a 95% probability that the true parameter lies within the interval.
T-statistic	A ratio of the departure of the estimated value of a parameter from its hypothesized value to its standard error. The degrees of freedom (df) are the number of independent values that can vary in an analysis without breaking any constraints.	The t-statistic directly tells us the probability of the null hypothesis being true.

4.7 Interpretation Questions

1. What does a significant p-value indicate in the context of this t-test?
2. How would you interpret the confidence interval in this analysis?

Show/Hide Solutions

Solutions:

1. **Significant p-value**: A significant p-value indicates that the observed data is unlikely under the null hypothesis. This suggests that there is evidence against the null hypothesis, implying a statistically significant difference between the satisfaction scores of the two groups. However, it does not measure the probability that the null hypothesis is true or false.
2. **Confidence interval**: The confidence interval provides a range of values that, based on the sample data, is likely to contain the true mean difference between the groups. If we were to repeat the experiment many times, we would expect a specified proportion (e.g., 95%) of these intervals to contain the true mean difference. It does not mean that there is a 95% probability that the true mean difference lies within this specific interval.

4.8 Comparison Tables

4.8.1 Assumptions and Statistical Analysis Commands

Step	R Command	Stata Command
Descriptive Statistics	`summary(data$satisfaction)`	`summarize satisfaction`
Box Plot	`ggplot(data, aes(x = group, y = satisfaction)) + geom_boxplot()`	`graph box satisfaction, over(group)`
T-Test	`t.test(satisfaction ~ group, data = data)`	`ttest satisfaction, by(group)`

4.8.2 Simulated Dataset Generation

Step	R Command	Stata Command
Generate Data	`rnorm(n, mean, sd)`	`rnormal(mean, sd)`
Save Data	`write.csv(data, "satisfaction_data.csv")`	`save satisfaction_data.dta, replace`

5 ANOVA

5.1 Introduction

This chapter covers ANOVA (Analysis of Variance), used to compare the means across multiple groups. We will use an example dataset to investigate whether the design of a user interface (UI) affects the time users spend on a website.

5.2 Example Question

Does the design of a user interface (UI) influence the time users spend on a website?

5.3 Required Packages (R)

```
# Load the necessary packages
library(tidyverse) # used for data manipulation and visualization
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.0     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(car) # provides tools for ANOVA and regression diagnostics
```

```
Loading required package: carData

Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode

The following object is masked from 'package:purrr':

    some
```

```
# to install any missing packages go to the Terminal and run the command: install.packages("
```

5.4 Simulating the Dataset in R

```
# Setting a seed for reproducibility
set.seed(123)

# Simulating data
n_groups <- 3  # Number of UI designs
n_per_group <- 50  # Number of users per group

# Creating a factor variable for UI design
ui_design <- factor(rep(1:n_groups, each = n_per_group))

# Simulating time spent data with different means for each UI design
time_spent <- rnorm(n_groups * n_per_group, mean = rep(c(20, 25, 22), each = n_per_group), s

# Creating a data frame
data <- data.frame(ui_design, time_spent)

# Viewing the first few rows of the dataset
head(data)
```

```
  ui_design time_spent
1         1   17.19762
2         1   18.84911
3         1   27.79354
```

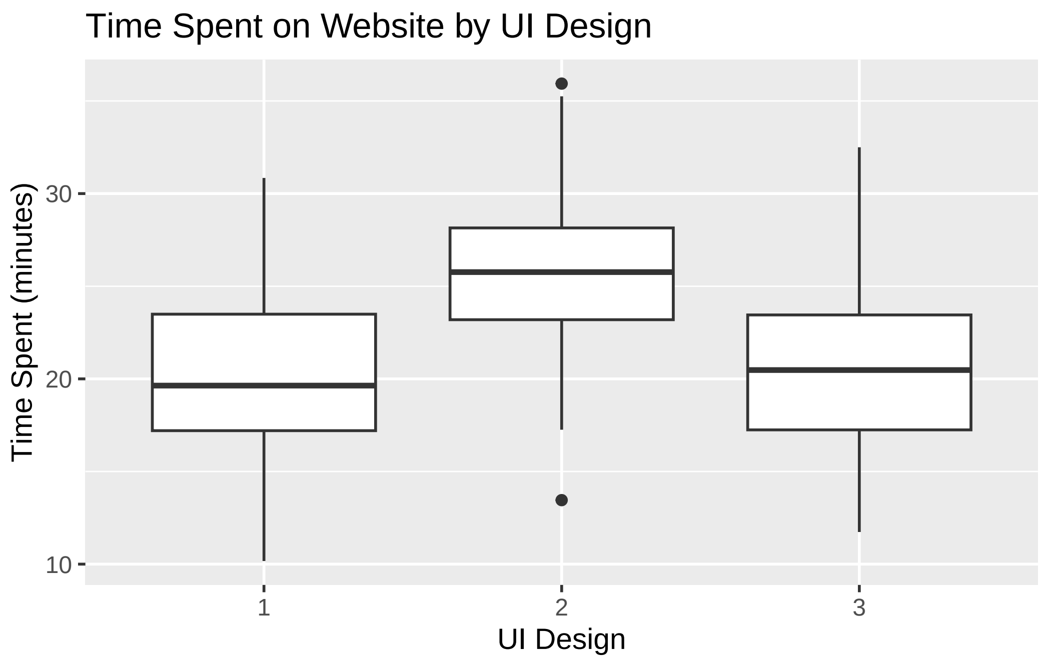
```
4          1   20.35254
5          1   20.64644
6          1   28.57532
```

5.5 Simulating the Dataset in Stata

```
* Set seed for reproducibility
set seed 123

* Simulate data
set obs 150
gen ui_design = ceil(_n/50)
gen time_spent = rnormal(20 + (ui_design==2)*5 + (ui_design==3)*2, 5)

* View the first few rows
list in 1/10
```

5.6 Visualizing the Descriptives in R

```
# Plotting the distribution of time spent across different UI designs
ggplot(data, aes(x = ui_design, y = time_spent)) +
  geom_boxplot() +
  labs(title = "Time Spent on Website by UI Design",
       x = "UI Design",
       y = "Time Spent (minutes)")
```



5.7 Visualizing the Descriptives in Stata

```
* Box plot of time spent by UI design
graph box time_spent, over(ui_design) title("Time Spent on Website by UI Design") ///
    ytitle("Time Spent (minutes)") xtitle("UI Design")
```

5.8 Running the ANOVA in R

```
# Performing ANOVA
anova_model <- aov(time_spent ~ ui_design, data = data)

# Viewing the summary of the ANOVA model
summary(anova_model)
```

```
             Df Sum Sq Mean Sq F value   Pr(>F)    
ui_design     2    937   468.7   21.18 8.29e-09 ***
Residuals   147   3253    22.1                     
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.9 Running the ANOVA in Stata

```
* Perform ANOVA
anova time_spent ui_design
```

5.10 Interpreting the Output

5.10.1 In R

The ANOVA table provides the following key pieces of information: - **Df**: Degrees of freedom associated with the sources of variance. - **Sum Sq**: Sum of squares, which measures the total variation for each source. - **Mean Sq**: Mean square, calculated as Sum Sq divided by Df. - **F value**: The F-statistic, calculated as the ratio of mean square values. - **Pr(>F)**: The p-value associated with the F-statistic.

5.10.2 In Stata

The output of the ANOVA in Stata provides similar information: - **Source**: Lists the sources of variance. - **Partial SS**: Partial sum of squares for each source. - **df**: Degrees of freedom associated with each source. - **MS**: Mean square for each source, calculated as SS/df. - **F**: The F-statistic for each source. - **Prob > F**: The p-value associated with the F-statistic.

If the p-value is less than the significance level (typically 0.05), we reject the null hypothesis that all group means are equal.

5.11 Post-hoc Testing in R

```
# Performing Tukey's Honest Significant Difference test
tukey_test <- TukeyHSD(anova_model)

# Viewing the Tukey test results
tukey_test
```

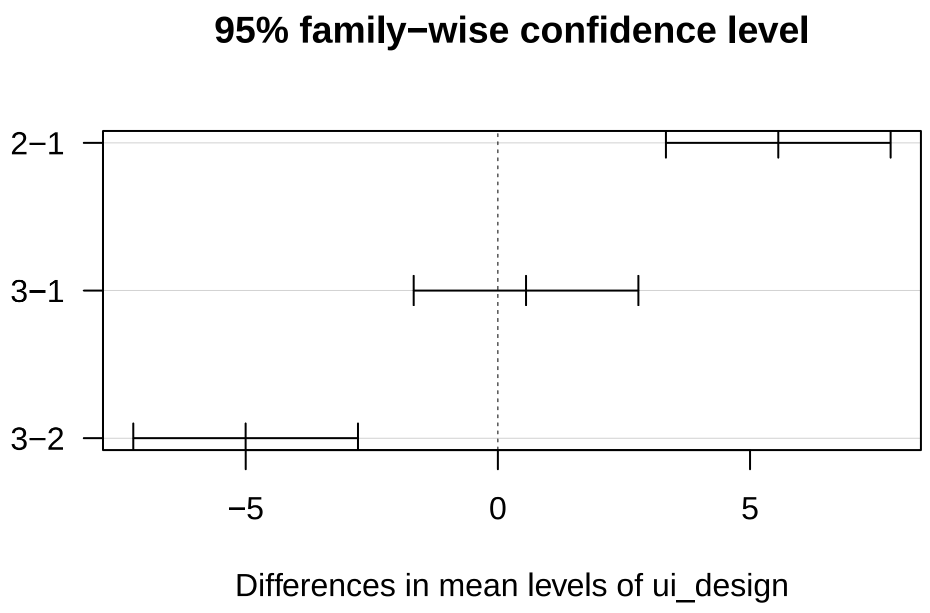
```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = time_spent ~ ui_design, data = data)

$ui_design
          diff       lwr       upr     p adj
2-1  5.5600236  3.332272  7.787775 0.0000001
3-1  0.5584801 -1.669271  2.786231 0.8237890
3-2 -5.0015435 -7.229295 -2.773792 0.0000012
```

5.12 Post-hoc Testing in Stata

```
* Perform Bonferroni post-hoc test
oneway time_spent ui_design, bonferroni
```

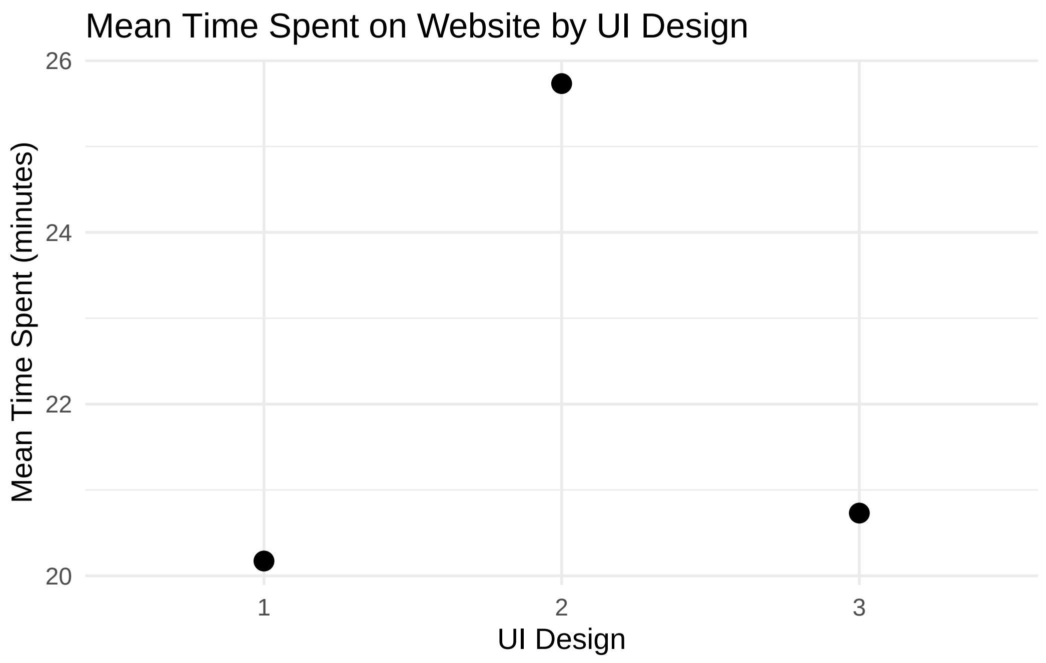
5.13 Plotting the Results in R

```
# Plotting the results of the Tukey HSD test
plot(tukey_test, las = 1)
```



```r
# Creating a plot to visualize group means with confidence intervals
ggplot(data, aes(x = ui_design, y = time_spent)) +
  stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.2) +
  stat_summary(fun = mean, geom = "point", size = 3) +
  labs(title = "Mean Time Spent on Website by UI Design",
       x = "UI Design",
       y = "Mean Time Spent (minutes)") +
  theme_minimal()
```

```
Warning: Computation failed in `stat_summary()`.
Caused by error in `fun.data()`:
! The package "Hmisc" is required.
```



5.14 Plotting the Results in Stata

```
* Plot group means with confidence intervals
means time_spent, over(ui_design) ci
```

5.15 Assumptions

- **Independence**: Observations should be independent of each other.
- **Normality**: The residuals of the model should be normally distributed.
- **Homoscedasticity**: Variances across the groups should be equal.
- **Random Sampling**: The data should be randomly sampled from the population.

These assumptions should be checked to ensure the validity of the ANOVA results.

5.16 Syntax Comparison: R vs Stata

This table summarizes the main differences between R and Stata in terms of syntax for performing ANOVA analyses.

Task	R Command	Stata Command
Simulating Data	`rnorm()` for simulating normal distribution	`rnormal()` for simulating normal distribution
Setting Seed for Reproducibility	`set.seed(123)`	`set seed 123`
Creating a Factor Variable	`factor()`	`gen variable` and `egen group`
Visualizing Descriptives	`ggplot()` with `geom_boxplot()`	`graph box`
Running ANOVA	`aov()` and `summary()`	`anova`
Post-hoc Testing	`TukeyHSD()`	`oneway` with `bonferroni` option
Plotting Group Means with Confidence Intervals	`ggplot()` with `stat_summary()`	`means` with `ci` option

6 Linear Regression

6.1 Introduction

This chapter covers how to perform linear regression to study the relationship between variables. We'll use an example dataset that simulates the relationship between study time and performance on an online learning platform.

6.2 Example Question

How does the amount of time spent on an e-learning platform (in hours) affect the test scores of users?

6.3 Dataset Simulation in R

```
# Load necessary package
set.seed(123)

# Simulate data
n <- 100
study_time <- rnorm(n, mean = 10, sd = 2)  # Average 10 hours
test_score <- 50 + 5 * study_time + rnorm(n, mean = 0, sd = 5)  # Linear relationship with so

# Create a data frame
data <- data.frame(study_time, test_score)

# View the first few rows
head(data)
```

6.4 Dataset Simulation in Stata

```
* Set seed for reproducibility
set seed 123

* Simulate data
set obs 100
gen study_time = rnormal(10, 2)
gen test_score = 50 + 5 * study_time + rnormal(0, 5)

* View the first few rows
list in 1/10
```

6.5 Performing Linear Regression

6.5.1 R

```
# Fit the linear regression model
model <- lm(test_score ~ study_time, data = data)

# View the summary
summary(model)
```

6.5.2 Stata

```
* Fit the linear regression model
regress test_score study_time
```

6.6 Assumptions

- **Linearity**: The relationship between the independent and dependent variable should be linear.
- **Independence**: Observations should be independent of each other.
- **Homoscedasticity**: The residuals should have constant variance at every level of the independent variable.
- **Normality**: The residuals should be normally distributed.

7 Multilevel Regression

7.1 Introduction

This chapter covers multilevel regression, where data is nested. We will explore how user satisfaction with a mobile app is affected by time spent on the app, considering that users are nested within different age groups.

7.2 Example Question

Does time spent on a mobile app influence user satisfaction, and does this effect differ across age groups?

7.3 Dataset Simulation in R

```r
# Load necessary package
set.seed(123)

# Simulate data
n_groups <- 5  # Number of age groups
n_per_group <- 50  # Number of users per group

age_group <- factor(rep(1:n_groups, each = n_per_group))
time_spent <- rnorm(n_groups * n_per_group, mean = 30, sd = 10)
satisfaction <- 3 + 0.2 * time_spent + as.numeric(age_group) + rnorm(n_groups * n_per_group,

# Create a data frame
data <- data.frame(age_group, time_spent, satisfaction)

# View the first few rows
head(data)
```

7.4 Dataset Simulation in Stata

```
* Set seed for reproducibility
set seed 123

* Simulate data
set obs 250
gen group = ceil(_n/50)  // Age group
gen time_spent = rnormal(30, 10)
gen satisfaction = 3 + 0.2 * time_spent + group + rnormal(0, 2)

* Convert group to a factor
egen group_factor = group(group)

* View the first few rows
list in 1/10
```

7.5 Performing Multilevel Regression

7.5.1 R

```
# Load necessary package
library(lme4)

# Fit the multilevel model
model <- lmer(satisfaction ~ time_spent + (1 | age_group), data = data)

# View the summary
summary(model)
```

7.5.2 Stata

```
* Fit the multilevel model
mixed satisfaction time_spent || group:
```

7.6 Assumptions

- **Normality of residuals**: The residuals at each level of the model should be normally distributed.
- **Linearity**: The relationship between predictors and the outcome should be linear at each level of the model.
- **Independence**: Observations within each group should be independent.
- **Homoscedasticity**: The variance of residuals should be consistent across all levels of the hierarchy.

8 Logistic Regression

8.1 Introduction

This chapter covers logistic regression, which is used when the outcome variable is binary. We will use an example dataset to investigate whether the frequency of technical support contact predicts whether a user continues to use a software product.

8.2 Example Question

Does the frequency of contacting technical support predict whether a user will continue using a software product?

8.3 Required Packages (R)

```
# Load the necessary packages
library(tidyverse) # used for data manipulation and visualization
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.0     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(broom) # for tidying the model output, making it easier to work with
library(Statamarkdown) # to run Stata commands in an R environment
```

```
Stata found at C:/Program Files/Stata18/StataBE-64.exe
The 'stata' engine is ready to use.
```

```
# Statamarkdown configuration
stataexe <- "C:/Program Files/Stata18/StataBE-64.exe" # Add your own path to the Stata execu
knitr::opts_chunk$set(engine.path=list(stata=stataexe))

# to install any missing packages go to the Terminal and run the command: install.packages("
```

8.4 Simulating the Dataset in R

```
# Setting a seed for reproducibility
set.seed(123)

# Simulating data
n <- 200
support_contact <- rpois(n, lambda = 2)  # Number of contacts with support
continued_use <- rbinom(n, size = 1, prob = 1 / (1 + exp(-(-1 + 0.5 * support_contact))))

# Creating a data frame
data <- data.frame(support_contact, continued_use)

# Viewing the first few rows of the dataset
head(data)
```

```
  support_contact continued_use
1               1             0
2               3             0
3               2             1
4               4             1
5               4             1
6               0             1
```

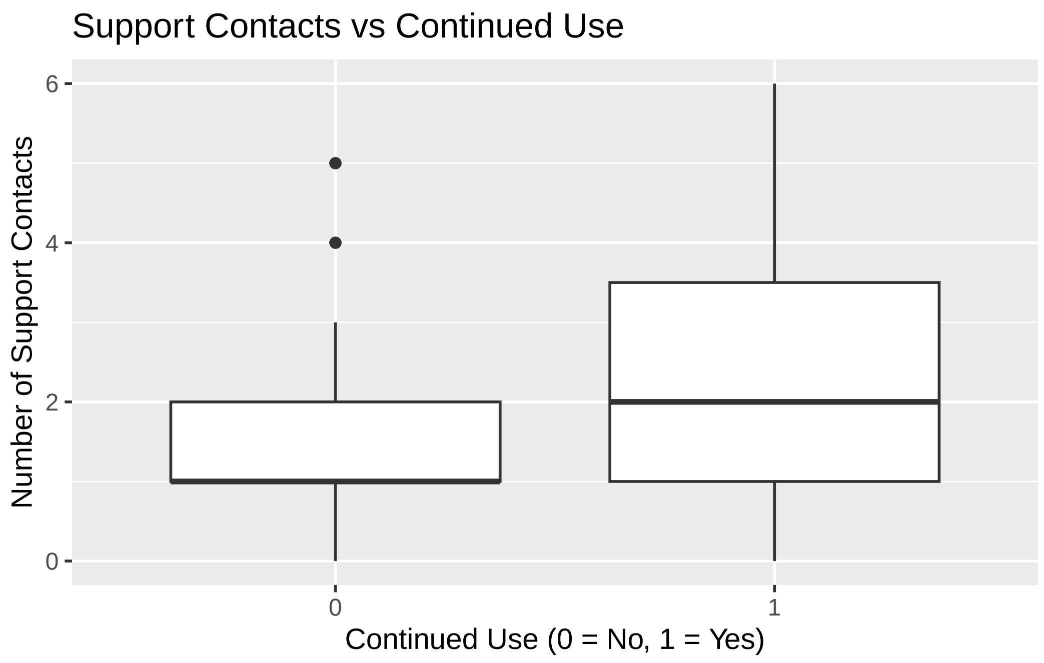
8.5 Simulating the Dataset in Stata

```
* Set seed for reproducibility
set seed 123

* Simulate data
set obs 200
gen support_contact = rpoisson(2)
gen continued_use = rbinomial(1, 1 / (1 + exp(-(-1 + 0.5 * support_contact))))

* Save to data file
save logreg_data.dta

* View the first few rows
list in 1/10
```

```
Number of observations (_N) was 0, now 200.



file logreg_data.dta already exists
r(602);

r(602);
```

8.6 Visualizing the Descriptives in R

```
# Plotting the distribution of support contacts for users who continued vs those who didn't
ggplot(data, aes(x = factor(continued_use), y = support_contact)) +
  geom_boxplot() +
  labs(title = "Support Contacts vs Continued Use",
       x = "Continued Use (0 = No, 1 = Yes)",
       y = "Number of Support Contacts")
```

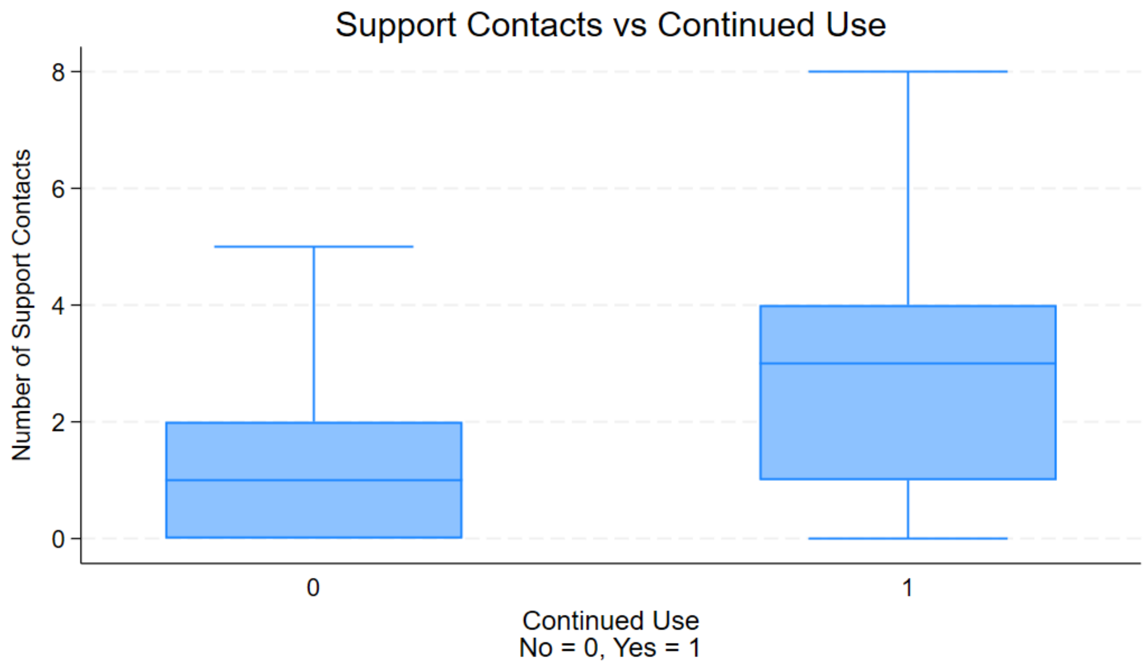


8.7 Visualizing the Descriptives in Stata

```
* NO NEED TO LOAD DATA AGAIN If USING STATA
use logreg_data.dta

* Data summary
summarize
```

```
    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
support_co~t |        200       1.875    1.523476          0          8
continued_~e |        200        .435    .4970011          0          1
```

```
* Box plot of support contacts by continued use
graph box support_contact, over(continued_use) title("Support Contacts vs Continued Use") b1
```



8.8 Running the Logistic Regression in R

```r
# Fitting the logistic regression model
logistic_model <- glm(continued_use ~ support_contact, data = data, family = "binomial")

# Viewing the summary of the logistic regression model
summary(logistic_model)
```

```
Call:
glm(formula = continued_use ~ support_contact, family = "binomial",
    data = data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)    
(Intercept)      -1.2023     0.3046  -3.947 7.90e-05 ***
support_contact   0.7398     0.1453   5.092 3.54e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 274.83  on 199  degrees of freedom
Residual deviance: 240.15  on 198  degrees of freedom
AIC: 244.15

Number of Fisher Scoring iterations: 4
```

8.8.1 Output interpretation

Term	Description
Coefficients	Estimates of the regression coefficients.
Std. Error	Standard errors of the coefficients.
z value	The test statistic for each coefficient.
Pr(>|z|)	The p-value associated with each coefficient, indicating whether it is statistically significant. If the p-value is less than the significance level (typically 0.05), we reject the null hypothesis that the coefficient is equal to zero.

8.9 Running the Logistic Regression in Stata

```
* Loading data to make it work in R environment, YOU DO NOT NEED TO LOEAD THE DATA AGAIN IN S
use logreg_data.dta

* Fit the logistic regression model
logit continued_use support_contact
```

```
>  DATA AGAIN IN STATA IF YOU ALREADY LOADED IT BEFORE!


Iteration 0:  Log likelihood = -136.93464
Iteration 1:  Log likelihood = -121.18683
Iteration 2:  Log likelihood = -121.17434
Iteration 3:  Log likelihood = -121.17434

Logistic regression                                     Number of obs =    200
                                                        LR chi2(1)    =  31.52
```

```
                                                    Prob > chi2   = 0.0000
Log likelihood = -121.17434                         Pseudo R2     = 0.1151

------------------------------------------------------------------------------
continued_~e | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
support_co~t |    .589038   .1166492     5.05   0.000     .3604098    .8176661
       _cons |  -1.379241   .2688518    -5.13   0.000    -1.906181   -.8523008
------------------------------------------------------------------------------
```

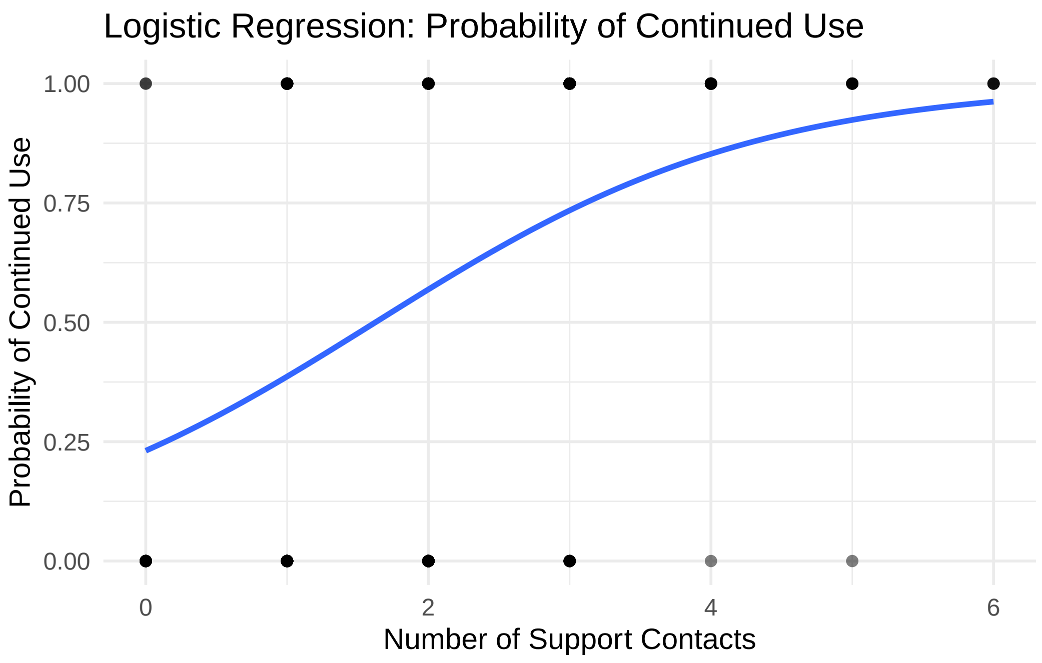
8.9.1 Output description

Term	Description
Coef.	Estimates of the regression coefficients.
Std. Err.	Standard errors of the coefficients.
z	The test statistic for each coefficient.
P>|z|	The p-value associated with each coefficient, indicating whether it is statistically significant. If the p-value is less than the significance level (typically 0.05), we reject the null hypothesis that the coefficient is equal to zero.

8.10 Plotting the Results in R

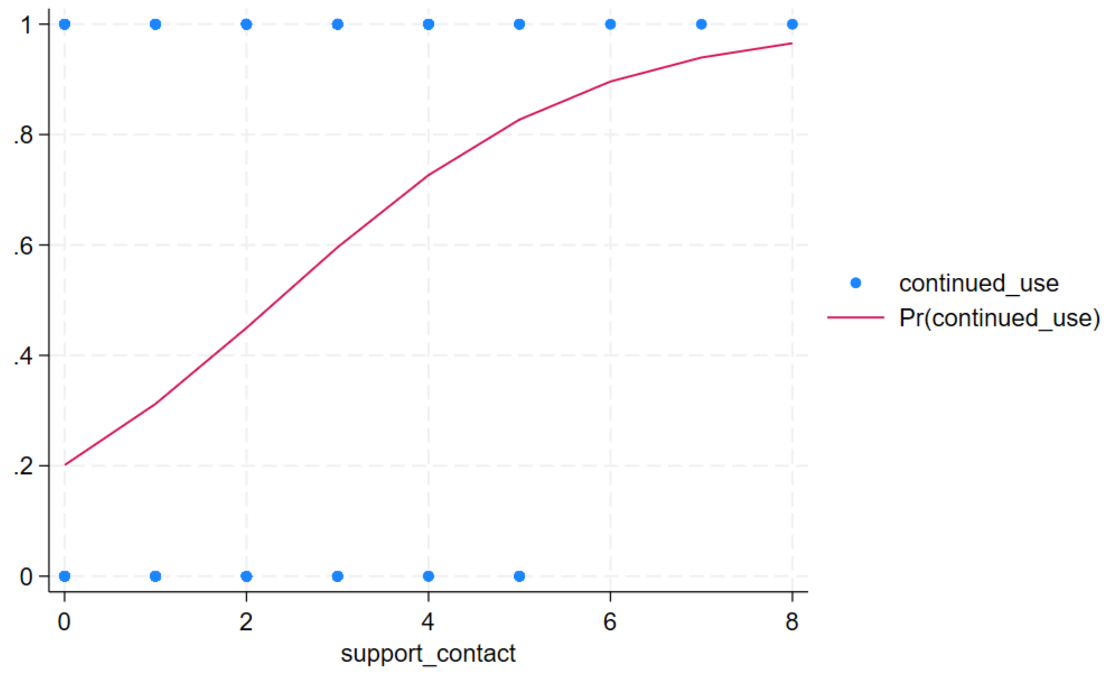
```r
# Plotting the logistic regression curve
ggplot(data, aes(x = support_contact, y = continued_use)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Logistic Regression: Probability of Continued Use",
       x = "Number of Support Contacts",
       y = "Probability of Continued Use") +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
```



8.11 Plotting the Results in Stata

```
* Create logistic regression plot

/* Generate the predicted probabilities
This command generates the predicted probabilities from the logistic regression model and sto

predict prob, pr

/* Sort the data by the predictor variable. Sorting the data by 'support_contact' ensures tha

sort support_contact

/* Plot the scatter plot with the logistic regression line This command creates a scatter plo

twoway (scatter continued_use support_contact) (line prob support_contact)
```



8.12 Assumptions

Assumption	Description
Binary Outcome	The dependent variable should be binary.
Independence	Observations should be independent of each other.
Linearity of logit	The logit (log-odds) of the outcome should be linearly related to the predictors.
No multicollinearity	The predictors should not be highly correlated with each other.
Large sample size	Logistic regression typically requires a large sample size to provide reliable estimates.

8.13 R

8.13.1 Binary oucome

```
table(data$continued_use)
```

```
  0   1
 89 111
```

8.13.2 Independence

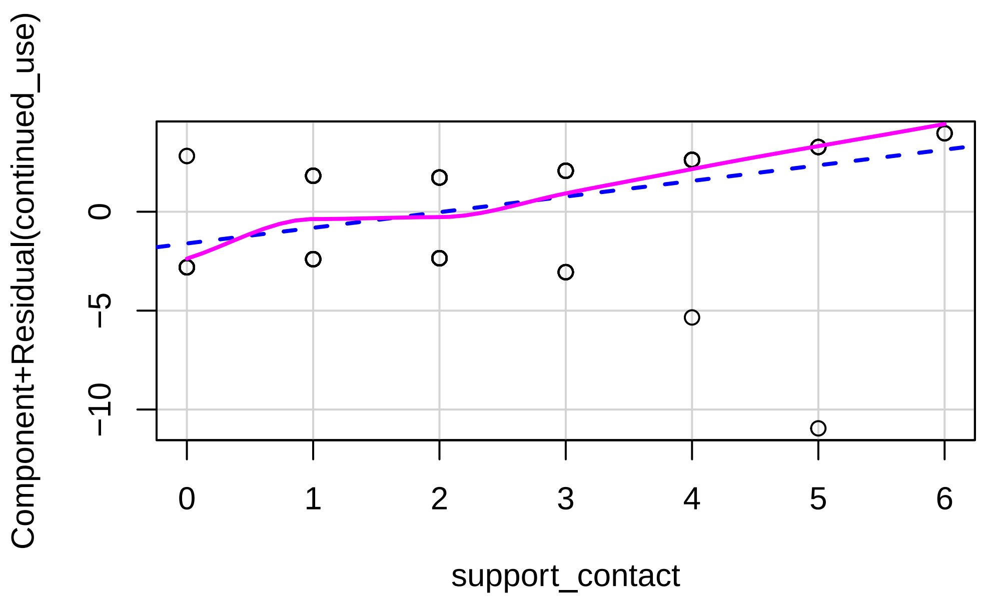
Verify that observations are independent. This is usually ensured by the study design.

8.13.2.1 Linearity of logit

```
library(car)
```

```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode

The following object is masked from 'package:purrr':

    some
```

```
logit_model <- glm(continued_use ~ support_contact, data = data, family = binomial)
crPlots(logit_model)
```



8.13.3 No Multicollinearity

Note: The assumption of no multicollinearity is relevant only when you have at least two predictors in your model. Multicollinearity occurs when two or more predictors are highly correlated with each other, which can make it difficult to determine the individual effect of each predictor on the dependent variable.

Check for multicollinearity among predictors.

```
vif(YOUR_MODEL_HER)
```

```
Error in eval(expr, envir, enclos): object 'YOUR_MODEL_HER' not found
```

8.13.4 Large sample size

Ensure you have a sufficiently large sample size. A rule of thumb is at least 10 events per predictor variable.

8.14 Stata

8.14.1 Binary oucome

```
tabulate continued_use
```

```
no variables defined
r(111);

r(111);
```

8.14.2 Independence

Verify that observations are independent. This is usually ensured by the study design.

8.14.3 Linearity of logit

```
gen logit_support_contact = log(support_contact / (1 - support_contact))
scatter logit_support_contact support_contact
```

```
support_contact not found
r(111);

r(111);
```

8.14.4 No Multicollinearity

```
estat vif
```

```
last estimates not found
r(301);

r(301);
```

8.14.5 Large sample size

Ensure you have a sufficiently large sample size. A rule of thumb is at least 10 events per predictor variable.

8.15 Syntax Comparison: R vs Stata

This table summarizes the main differences between R and Stata in terms of syntax for performing Logistic Regression analysis.

Task	R Command	Stata Command
Simulating Data	`rpois()`, `rbinom()`	`rpoisson()`, `rbinomial()`
Setting Seed for Reproducibility	`set.seed(123)`	`set seed 123`
Visualizing Descriptives	`ggplot()` with `geom_boxplot()`	`graph box`
Running Logistic Regression	`glm()` with `family = "binomial"`	`logit`
Plotting the Results	`ggplot()` with `geom_smooth(method = "glm", ...)`	`twoway scatter` and `lfit`

9 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```