Regression Analysis With R And Easystats

Author

Maulik Bhatt

S P Jain Institute of Management & Research
Mumbai

2023
Maulik Bhatt

Contents

List of Figures

List of Tables

Preface

Acknowledgement

I would like to express my heartfelt gratitude to the individuals who have played a significant role in the creation of this book on Regression Analysis with R. Their guidance, knowledge, and support have been invaluable throughout this journey.

First and foremost, I extend my deepest appreciation to my esteemed professors, [Professor's Name], [Professor's Name], and [Professor's Name]. Their expertise in statistics and econometrics has shaped my understanding of quantitative analysis and provided a solid foundation for this book. Their dedication to teaching and their unwavering commitment to fostering academic excellence have been instrumental in my growth as a student of econometrics.

I would like to extend special thanks to my colleague and dear friend, [Colleague's Name]. Their introduction to the world of R programming language opened up a world of possibilities for me. Their patience, willingness to share their knowledge, and countless hours spent assisting me with R-related challenges have been indispensable. I am grateful for their unwavering support and the collaborative environment we fostered, which significantly enriched my learning experience.

A heartfelt appreciation goes to the developers of R, an open-source programming language that has revolutionized the field of data analysis and statistical computing. Their tireless efforts in

creating and continuously improving R have made it an indispensable tool for researchers and analysts worldwide. Without their dedication, this book would not have been possible.

I would also like to express my gratitude to the developers of the R packages that have been instrumental in the analysis and visualization techniques presented in this book. Their commitment to excellence, innovation, and user-friendly implementations has immensely contributed to the field of regression analysis. Their packages have not only expanded the capabilities of R but have also facilitated the seamless integration of econometric methodologies into practical applications.

Finally, I extend my heartfelt thanks to my family and friends who have supported me throughout this writing process. Their encouragement, understanding, and belief in my abilities have been a constant source of motivation.

To all those mentioned above, and to anyone else who has contributed to this book in any way, I offer my deepest appreciation. Your guidance, knowledge, and support have been invaluable in shaping this work and have helped me fulfill my goal of sharing the beauty and importance of regression analysis with R.

Thank you.

[Your Name]

1.

Introduction

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Importance of regression analysis in econometrics*
- *Overview of the book's structure and goals*
- *Introduction to R programming language and its relevance to econometric analysis*

Econometrics is a combination of two words: Econo + Metric. Econo refers to concepts of economics, while metric refers to measurement. Let's take an example of the law of demand from microeconomic theory. We know the demand of a commodity will decrease if the price of the commodity increases. But we don't know how much the demand will decrease, given the increase in the price is one

unit (i.e., one Dollar, or one Euro, etc.). In Econometrics, we measure such increase or decrease using various experiments. In these experiments, we observe how much the demand for a commodity increase of decrease in response to a unit change (increase or decrease) in price of the commodity.

1.1. Importance of R for Econometrics and Statistics

Many software are available for statistical and econometric analysis. Some of the popular software include SPSS, SAS, Stata, MATLAB, etc. These are all proprietary software. Free and open source alternatives include python and R, which are programming languages. The users need to type the commands in order to perform various tasks/analysis in these programming languages.

Python is a general purpose programming langauge, while R is a statistical programming language. R also has many packages (i.e., add-ons, which enhance the functionality of R). As of writing this, there are `19697` packages available on CRAN (the Comprehensive R Archive Network, which is the central authority to decide about R programming language). Even more packages are available on `github`, which is an Internet hosting service for software development and version control using the version control system Git. Because of such extensive support, I have chosen R to conduct econometric analysis for this book.

In particular, I will use `easystats` set of packages, which are designed for various statistical analysis tasks. We will see how `easystats` make our tasks easier in R.

I will also use `quarto`, which is a document preparation system, based on pandoc's markdown. The benefit of `quarto` is that we can keep our analysis and prose in the same document. At the end, when we click on the `build` button, it renders the entire book in either pdf or HTML format. It is also possible to render the book in Microsoft Word format, but this output format has got limited

support. It is more convenient to design and customize the output in HTML and PDF formats. We will discuss more about this in the chapter of Getting Started with R.

2.

Introduction to Econometrics and Regression Analysis

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Understanding the basics of econometrics*
- *Role of regression analysis in econometric modeling*
- *Overview of the regression analysis process*
- *Introduction to R for econometric analysis*

Basics of Econometrics

Role of Regression

Introduction to R

3.

Getting Started with R for Econometrics

Status
This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Introduction to R programming language and its ecosystem*
- *Setting up the R environment and installing necessary packages*
- *Loading and manipulating data in R for econometric analysis*
- *Exploring data visualization techniques using R*

3.1. A Brief History of R

John Chambers, working at Bell Laboratories, developed S programming language for statistical analysis. This programming language was incorporated in the commercial program S-plus. Inspired by this programming languagage, tw o statistics professors Robert Gentleman and Ross Ihaka developed a reduced version of S, which they named R.

R was first released in 1993. In 1995, another statistician Martin Mächler convinced Gentleman and Ihaka to make R source code available to general public. As a result, R became free and open source under GNU General Public License.

R also has a robust community of users. As of writing this, there are 19,719 user-developed packages on Comprehensive R Archive Network (CRAN), which distributes the R source code and packages. Apart from CRAN, many more packages are also available on code sharing platform github. Users can download packages from github as well. The Appendix A explains how to install packages from CRAN and from github.

How to Install R

You can install R from the website of CRAN. Downloadable files and instructions to download and install R for all the major operating systems are available on this website. R can be downloaded and installed on Windows, Mac and Linux platforms.

It is possible to use R right from within this R Graphical User Interface (R-Gui). But such a work will not be reproducible. It means, you will be able to work on your system, but you won't be able to share that work with others. IDE like RStudio makes it much easier to work in R, and to share your work with others.

How to Install RStudio

Once you have installed R for your operating system, you can visit the website of RStudio. Just like R, RStudio is also available on Windows, Mac and Linux platforms.

Useful Settings in RStudio

I would recommend you not to save the workspace in RStudio. This setting can be disabled in RStudio as follows:

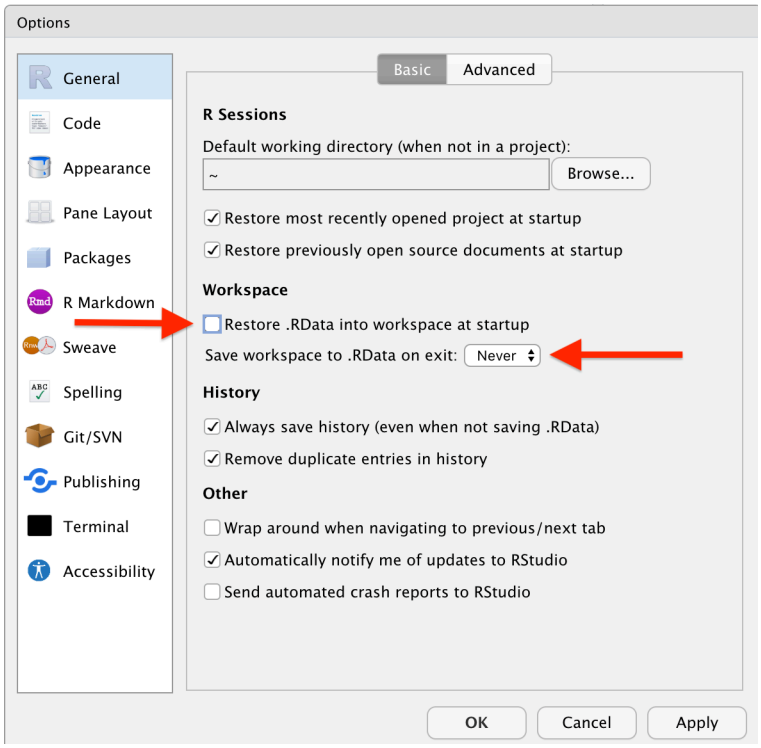

Figure 3.1.: How to Disable Saving Workspace

This setting (of not saving your workspace) seems counter-intuitive. But it helps you in the longer term. Assume you have to share your work with someone else. The other person might not have the same settings of computer in his/her system. So, the work on your system might look different in that system. Instead, you should save the R Script (or even better, a Quarto document) in your system, and should share it.

The Concept of Working Directory

R has a concept of working directory. When you save your R script and your work, it gets saved in the "working directory" of R. You can find and change the working directory using R commands.

```
# Find the working directory
getwd()
#Change the working directory
setwd("Path/to/another/folder")
```

3.2. Quarto

Quarto is a new open-source scientific and technical publishing system developed by Posit (the maker of RStudio). Quarto is designed to be useful to anyone who wants to create reproducible documents. A Quarto document contains both - the code and the prose. For example, if you are running regression analysis in R, R Script will only contain your code, but a Quarto document will contain your R code and your interpretation of the regression results. Quarto runs computations into separate pluggable language "engines", which helps make this cross-language functionality easier to support .

Here are some points that emphasize the reproducibility of Quarto over R scripts:

1. **Cross-language support**: Quarto is designed to work with multiple languages, including Python, bash, Julia, C, SQL, and more. This makes it easier to work with different languages in the same document.

2. **Built-in output formats**: Quarto generates the output in various formats like Microsoft Word, HTML, PDF, beamer, revealjs, etc. It also has many options for customizing each format.

3. **Native features for special project types**: Quarto has native features for special project types like websites, books, and blogs. This means that you don't have to rely on external packages. As a matter of fact, this book is written entirely using Quarto in RStudio.

4. **Easier rendering**: Quarto isn't an R package. It's a command-line interface that makes it much easier to work with Quarto documents outside of the RStudio IDE. You can also use Quarto in other IDEs like VS Code.

These features make Quarto a better choice than R scripts when it comes to reproducibility. With Quarto, you can easily create documents that are easy to reproduce and share with others.

Optional: Installing LaTeX

If you want to work in Quarto, and want to generate PDF output, LaTeXis required. There are two popular LaTeXdistributions: MiKTeX and TeX Live. However, I prefer another LaTeXdistribution, called TinyTeX. It's a light weight distribution of LaTeX, and works well with R. You can install it using R as follows:

```
#install tinytex R package
install.packages("tinytex")
#load it in R
library(tinytex)
#use this package to install LaTeX compiler TinyTeX
install_tinytex()
```

These commands install the most commonly used LaTeXpackages into your system. However, if you want to install all the LaTeXpackages, you can do so by using `install_tinytex(bundle = "TinyTeX-2")` instead of `install_tinytex()` function above.

3.3. Your First Interaction with R

After installing R and RStudio, you would start RStudio (not R). RStudio has four parts in the screen. The four parts (called "panes") are source pane, console pane, environment pane and files pane. You can customize each pane using `Tools Menu > Global Options > Pane Layout`. The RStudio screen looks as follows:

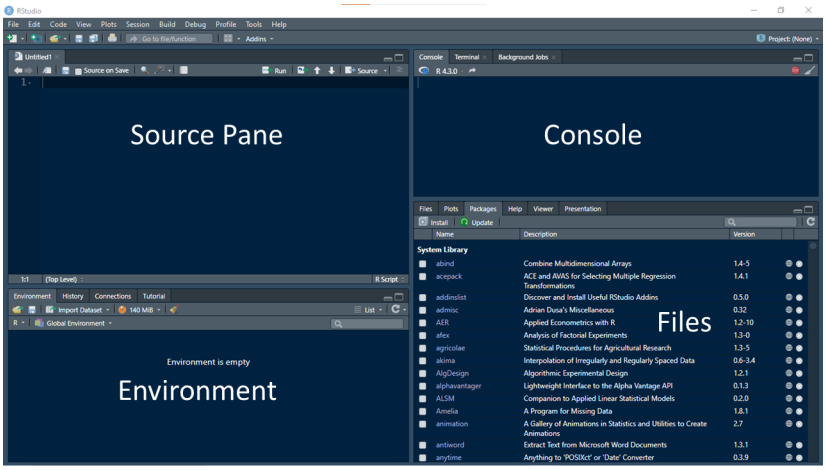

Figure 3.2.: RStudio Panes

When you open a new file in RStudio using the `Files Menu > New File`, you are presented with many options like R Script, R Markdown Document, Quarto Document, etc. I would suggest you to choose R Markdown Document or Quarto Document, because in these two file formats, you can write both - your analysis and your prose. Your document will be saved in your working directory.

In RStudio's File Menu, you will also find another option: `New Project`. This is a considerable improvement over a new file - be it an R Script or a Quarto Document. This is because it changes the working directory of the project to the folder, where `.Rproj` file of the project is stored. You can share this folder with anyone, and the code will work just fine on that computer also.

When you go to `File Menu > New Project > New Directory`, you get options as follows:

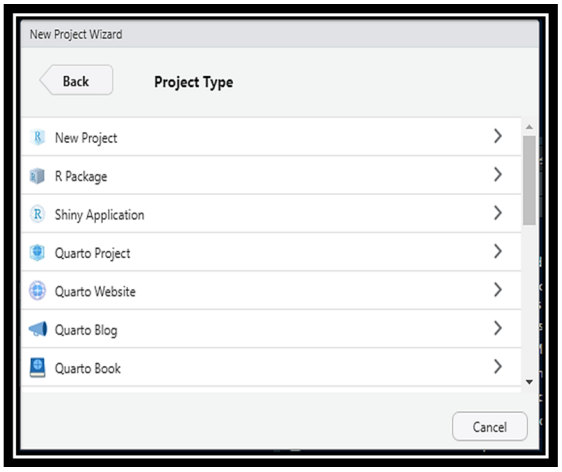

Figure 3.3.: Types of RStudio Projects

As you can see, it has many options like RStudio Project, R Package, Quarto Project, Quarto Website, Quarto Blog and Quarto Book. This book is written using a Quarto Book project.

3.4. Installing Packages

You can install packages using the command `install.packages(package_name)`. For example, the command to install the `easystats` set of packages would be `install.packages(easystats)`.

3.5. Loading and Manipulating Data

In order to analyse the data, we need to load or import the data first. There are many possibilities as far as loading/importing data is concerned. If the data comes from a package, then you can directly load the data using the `data(dataset_name)` command. For example, the command `data(mtcars)` will load the `mtcars` dataset into the environment. This dataset comes in-built in R.

Most of the data comes from external sources in different file formats like csv or Microsoft Excel. Data in csv file format can easily be read using the `read.csv(path_to_data_file)` command.

There is `readr` package also, which is part of the `tidyverse` set of packages. This package `readr` can import csv data and load it in the environment as a `tibble`. The native data format in R is `data.frame`, and `tibble` is "advanced version of `data.frame` according to the developers of the tidyverse set of packages.

Similarly, to import data from Microsoft Excel file, there is `readxl` package, which is also a part of `tidyverse`. Just like `readr`, `readxl` would also import the data in form of a `tibble`. Both the packages also share similar interface to import data.

The packages `foreign` and `haven` are useful for importing data from other statistical packages like SPSS, SAS and STATA.

Another noteworthy package is `data.table`, which can import data from many formats. Its function `fread` is versatile and can read data in many formats.

3.5.1. Manipulating Data

Although R is capable of data manipulation without using any other packages, some of the packages created especially to facilitate data manipulation makes the process easy and understandable for others. In our examples, we will use `tidyverse` set of packages for data manipulation. In particular, two of its packages `dplyr` and `tidyr` make this process quite easy.

3.5.2. Selecting Columns from Dataset

We can use the `select` command from `dplyr` for selecting columns from a dataset.

```
1 library(tidyverse)
2 mtcars |> select(am:carb) |> knitr::kable()
```

Table 3.2.: Selecting Columns from am to carb from mtcars dataset

	am	gear	carb
Mazda RX4	1	4	4
Mazda RX4 Wag	1	4	4
Datsun 710	1	4	1
Hornet 4 Drive	0	3	1
Hornet Sportabout	0	3	2
Valiant	0	3	1
Duster 360	0	3	4
Merc 240D	0	4	2
Merc 230	0	4	2
Merc 280	0	4	4
Merc 280C	0	4	4
Merc 450SE	0	3	3
Merc 450SL	0	3	3
Merc 450SLC	0	3	3
Cadillac Fleetwood	0	3	4
Lincoln Continental	0	3	4
Chrysler Imperial	0	3	4
Fiat 128	1	4	1
Honda Civic	1	4	2
Toyota Corolla	1	4	1
Toyota Corona	0	3	1

	am	gear	carb
Dodge Challenger	0	3	2
AMC Javelin	0	3	2
Camaro Z28	0	3	4
Pontiac Firebird	0	3	2
Fiat X1-9	1	4	1
Porsche 914-2	1	5	2
Lotus Europa	1	5	2
Ford Pantera L	1	5	4
Ferrari Dino	1	5	6
Maserati Bora	1	5	8
Volvo 142E	1	4	2

In the above command, we first loaded the `tidyverse` set of packages, so that its functionality is available in the R session. Then we took the built-in dataset `mtcars`, and then selected the columns from `am` to `carb` in the dataset. The `|>` is called the `pipe operator` in R. It was introduced in R version 4.1. Before that, there was another pipe operator (`%>%`) in the `magrittr` package, which is also a part of `tidyverse`. To understand how pipe operator makes coding easy and readable, imagine you want to take g(f(x)), which reads g of f of x. It means you want to take a value x, apply the function f on that value, and on the output, you want to apply another function g. Before the introduction of pipe operator, you would have written it as `g(f(x))`, but now after R 4.1, you can write it as `x |> f() |> g()`.

3.5.3. Selecting Rows from a Dataset (aka `filtering`)

We can use `filter` function from `dplyr` for filtering rows of a dataset. For example, we want to filter the rows of `mtcars` dataset, where `cyl` is equal to 6. We can do this by

```
mtcars |> filter(cyl==6) |> knitr::kable()
```

Table 3.3.: Selecting Rows from mtcars dataset where value of cyl is 6

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6

We can see that in this code, we have used "=" twice. In R, "=" is an assignment operator. It means, when we write `cyl=6`, it assigns value 6 to the variable `cyl`. So, when we want to check equality, we have to use "==" instead of "=".

3.5.4. Arrange Data

Using `dplyr`, it is easy to arrange or sort data in ascending or descending order of a column: use the `arrange` function for that.

```
mtcars |> arrange(mpg)|> select(mpg:disp) |> knitr::kable()
```

Table 3.4.: Arrange the Data in Ascending Order of `mpg`

	mpg	cyl	disp
Cadillac Fleetwood	10.4	8	472.0
Lincoln Continental	10.4	8	460.0
Camaro Z28	13.3	8	350.0
Duster 360	14.3	8	360.0
Chrysler Imperial	14.7	8	440.0
Maserati Bora	15.0	8	301.0
Merc 450SLC	15.2	8	275.8
AMC Javelin	15.2	8	304.0
Dodge Challenger	15.5	8	318.0
Ford Pantera L	15.8	8	351.0
Merc 450SE	16.4	8	275.8
Merc 450SL	17.3	8	275.8
Merc 280C	17.8	6	167.6
Valiant	18.1	6	225.0
Hornet Sportabout	18.7	8	360.0
Merc 280	19.2	6	167.6
Pontiac Firebird	19.2	8	400.0
Ferrari Dino	19.7	6	145.0
Mazda RX4	21.0	6	160.0

	mpg	cyl	disp
Mazda RX4 Wag	21.0	6	160.0
Hornet 4 Drive	21.4	6	258.0
Volvo 142E	21.4	4	121.0
Toyota Corona	21.5	4	120.1
Datsun 710	22.8	4	108.0
Merc 230	22.8	4	140.8
Merc 240D	24.4	4	146.7
Porsche 914-2	26.0	4	120.3
Fiat X1-9	27.3	4	79.0
Honda Civic	30.4	4	75.7
Lotus Europa	30.4	4	95.1
Fiat 128	32.4	4	78.7
Toyota Corolla	33.9	4	71.1

You can similarly arrange the data in descending order of `mpg` column in the `mtcars` dataset.

```
mtcars |> arrange(desc(mpg)) |> select(mpg:disp) |> knitr::kable()
```

Table 3.5.: Arrange the Data in Descending Order of `mpg`

	mpg	cyl	disp
Toyota Corolla	33.9	4	71.1
Fiat 128	32.4	4	78.7

	mpg	cyl	disp
Honda Civic	30.4	4	75.7
Lotus Europa	30.4	4	95.1
Fiat X1-9	27.3	4	79.0
Porsche 914-2	26.0	4	120.3
Merc 240D	24.4	4	146.7
Datsun 710	22.8	4	108.0
Merc 230	22.8	4	140.8
Toyota Corona	21.5	4	120.1
Hornet 4 Drive	21.4	6	258.0
Volvo 142E	21.4	4	121.0
Mazda RX4	21.0	6	160.0
Mazda RX4 Wag	21.0	6	160.0
Ferrari Dino	19.7	6	145.0
Merc 280	19.2	6	167.6
Pontiac Firebird	19.2	8	400.0
Hornet Sportabout	18.7	8	360.0
Valiant	18.1	6	225.0
Merc 280C	17.8	6	167.6
Merc 450SL	17.3	8	275.8
Merc 450SE	16.4	8	275.8
Ford Pantera L	15.8	8	351.0
Dodge Challenger	15.5	8	318.0

	mpg	cyl	disp
Merc 450SLC	15.2	8	275.8
AMC Javelin	15.2	8	304.0
Maserati Bora	15.0	8	301.0
Chrysler Imperial	14.7	8	440.0
Duster 360	14.3	8	360.0
Camaro Z28	13.3	8	350.0
Cadillac Fleetwood	10.4	8	472.0
Lincoln Continental	10.4	8	460.0

3.5.5. Create a New Variable

You can create a new variable using the `mutate` function in `dplyr` package. For example, you want to create a new variable, called FuelEff, which is defined as reciprocal of mpg, then you can do it as follows:

```
1 mtcars |> mutate(FuelEff = 1/mpg) |> select(am:FuelEff) |> knitr::kable()
```

Table 3.6.: Create a New Variable in `mtcars` dataset

	am	gear	carb	FuelEff
Mazda RX4	1	4	4	0.0476190
Mazda RX4 Wag	1	4	4	0.0476190
Datsun 710	1	4	1	0.0438596
Hornet 4 Drive	0	3	1	0.0467290

	am	gear	carb	FuelEff
Hornet Sportabout	0	3	2	0.0534759
Valiant	0	3	1	0.0552486
Duster 360	0	3	4	0.0699301
Merc 240D	0	4	2	0.0409836
Merc 230	0	4	2	0.0438596
Merc 280	0	4	4	0.0520833
Merc 280C	0	4	4	0.0561798
Merc 450SE	0	3	3	0.0609756
Merc 450SL	0	3	3	0.0578035
Merc 450SLC	0	3	3	0.0657895
Cadillac Fleetwood	0	3	4	0.0961538
Lincoln Continental	0	3	4	0.0961538
Chrysler Imperial	0	3	4	0.0680272
Fiat 128	1	4	1	0.0308642
Honda Civic	1	4	2	0.0328947
Toyota Corolla	1	4	1	0.0294985
Toyota Corona	0	3	1	0.0465116
Dodge Challenger	0	3	2	0.0645161
AMC Javelin	0	3	2	0.0657895
Camaro Z28	0	3	4	0.0751880
Pontiac Firebird	0	3	2	0.0520833
Fiat X1-9	1	4	1	0.0366300

	am	gear	carb	FuelEff
Porsche 914-2	1	5	2	0.0384615
Lotus Europa	1	5	2	0.0328947
Ford Pantera L	1	5	4	0.0632911
Ferrari Dino	1	5	6	0.0507614
Maserati Bora	1	5	8	0.0666667
Volvo 142E	1	4	2	0.0467290

Summarize a Data

Suppose you want to see what is the average `mpg` in the whole dataset? Remember mpg stands for miles per gallon. So, this column shows the fuel efficiency of various cars. So, you can use `summary` function from dplyr to get this answer.

```
mtcars |> summarise("Average MPG" = mean(mpg)) |> knitr::kable()
```

Table 3.7.: Average Miles Per Gallon of Cars in USA

Average MPG
20.09062

However, this function `summarise` becomes more powerful when combined with another function `group_by`. For example, you want to know if there is difference in mean mpg for different level of cyl. You can see it using the following code:

```
mtcars |> group_by(cyl) |> summarise("Average MPG" = mean(mpg)) |> knitr::kable()
```

Table 3.8.: Miles Per Gallon for Different Values of Cylinder

cyl	Average MPG
4	26.66364
6	19.74286
8	15.10000

4.

Simple Linear Regression

Status
This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Understanding the principles of simple linear regression*
- *Performing simple linear regression in R for econometric analysis*
- *Interpreting regression results in the context of economic variables*
- *Assessing model assumptions and addressing violations*
- *Practical examples and exercises using R*

```
library(easystats)
library(ggplot2)
```

Introduction

Linear regression is among the fundamental concepts in Econometrics. In linear regression, we try to estimate how much one variable will change, with response to a change in another variable. The controlled variable is called predictor or independent variable. The variable, which changes as a response to a change in the controlled variable is called response or dependent variable. In R, we denote such relationship with `y~x`, where y is the response and x is the predictor. The notation `y~x` is read as "y explained by x".

In mathematical terms, we are fitting a linear equation between x and y. When we write `y~x`, it means `y = ax + b`, and we need to find a and b from the data points given.

Let's understand what it means for us. The workflow for the linear regression problem would be:

1. We would be given some observations of both - independent variable and dependent variable.
2. we graph these data points, using a coordinate system (like Cartesian system). Each value is represented by a dot. Such a diagram is called a scatter-plot diagram.
3. After graphing the points onto a scatter plot diagram, linear regression analysis seeks to find the best-fit line to fit the points as closely as possible.
4. This best-fit line is a line, which minimizes the distance between the points falling above or below the lines.

Linear Regression 101: What Is It And How Is It Done? | Fiverr

Principles of Linear Regression (Gauss Markov)

Example in R

For understanding regression, we can generate a dummy dataset, and create a regression model on the basis of the data.

```r
dummy <- data.frame(y = rnorm(100),x = rpois(100,3))
```

In the code above, we have created a dummy data. The `rnorm(n)` function generates n number of data points based on normal distribution. The default mean and standard deviation are 0 and 1 respectively. Similarly, the `rpois(n,l)` function generates n number of data points based on poisson distribution, with = l. We can plot the data to explore it visually. We will use `ggplot2` package to create this scatterplot diagram.

```r
ggplot(dummy)+aes(x,y)+geom_point()+labs(title = "Scatterplot Diagram of X and Y")
```

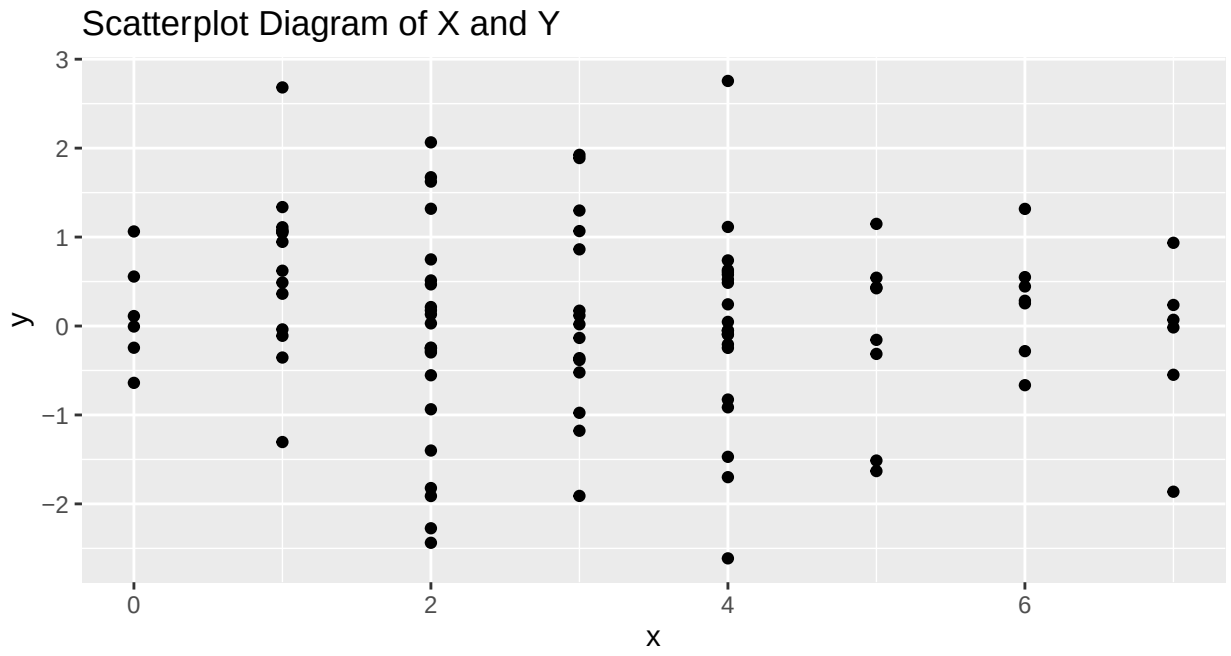

Figure 4.1.: Scatterplot of X and Y

Now we can create a regression model from this dataset.

```
model1 <- lm(y~x, data = dummy)
```

This model stores a lot of information, but it is difficult to understand and get the meaning out of it. So, one way is to use the `summary` function on this `model1` object.

```
summary(model1)
```

```
Call:
lm(formula = y ~ x, data = dummy)

Residuals:
    Min      1Q  Median      3Q     Max
-2.6342 -0.4445  0.0324  0.6090  2.7336

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.24693    0.20632   1.197    0.234
x           -0.05627    0.05672  -0.992    0.324

Residual standard error: 1.051 on 98 degrees of freedom
Multiple R-squared:  0.009944,  Adjusted R-squared:  -0.0001589
F-statistic: 0.9843 on 1 and 98 DF,  p-value: 0.3236
```

The problem with this summary object is that it is not a data.frame. It is difficult to put this object in a research paper or any other academic submisison. So, we can use the `easystats` packages

for this. The `model_parameters` function would show us the parameters (aka coefficients) of the model, and the `model_performance` function will show us the effectiveness of the regression model. We can also use the `display` function to beautify the table in the output.

```
model1 |> model_parameters() |>
  display(format = "markdown", caption = "Regression Parameters")
```

Table 4.2.: Regression Parameters

Parameter	Coefficient	SE	95% CI	t(98)	p
(Intercept)	0.25	0.21	(-0.16, 0.66)	1.20	0.234
x	-0.06	0.06	(-0.17, 0.06)	-0.99	0.324

```
model1 |> model_performance() |>
  display(format = "markdown", caption = "Regression Effectiveness")
```

Table 4.3.: Regression Effectiveness

AIC	AICc	BIC	R2	R2 (adj.)	RMSE	Sigma
297.71	297.96	305.53	9.94e-03	-1.59e-04	1.04	1.05

4.0.1. Interpretation of Regression Model

- The intercept is statistically non-significant and negative (beta = -0.16, 95% CI [-0.59, 0.27], t(98) = -0.74, p = 0.461; Std. beta = -5.50e-18, 95% CI [-0.20, 0.20]).

- The effect of x is statistically non-significant and positive (beta = 0.07, 95% CI [-0.04, 0.17],

t(98) = 1.24, p = 0.219; Std. beta = 0.12, 95% CI [-0.07, 0.32]).

The model explains a statistically not significant and very

weak proportion of variance (R^2 = 0.02, F(1, 98) = 1.53, p =

0.219, adj. R^2 = 5.32e-03).

4.0.2. Non-Linear Relationship in Regression

Sometimes, the relationship between the dependent and independent variable is not linear, but quadratic. This can be understood either from the scatter plot diagram or from theoretical considerations. In mathematical terms, the relationship is

$$Y = \beta_0 + \beta_1 * X^2 + \gamma \tag{4.1}$$

where X and Y are independent and dependent variables, respectively. This is still considered "Linear Regression", because we are interested in the the linearity of the coefficient term (i.e.,), not the linearity of the variables. In the equation above, both the coefficients (β_0 and β_1) are linear, and so is the error term, γ in this equation.

5.

Multiple Linear Regression

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Extending regression analysis to multiple independent variables*
- *Building and interpreting multiple linear regression models in R*
- *Handling multicollinearity and selecting significant predictors in an economic context*
- *Model evaluation and diagnostics in econometric regression*
- *Application of multiple linear regression in economic analysis using R*

5.0.1. Introduction

In the previous chapter, we saw simple linear regression, in which we attempted to understand the relationship between two variables - one called independent variable and the other called the dependent variable. Multiple linear regression is a statistical technique that allows us to analyze the relationship between two or more independent variables and a dependent variable. It is an extension of simple linear regression, which only involves one independent variable. Multiple linear regression is used to predict the value of a dependent variable based on the values of two or more independent variables. It is commonly used in fields such as economics, finance, and social sciences.

Assumptions of Multiple Linear Regression

There are a few assumptions of multiple linear regression. They are as follows:

1. Linearity of coefficients and error terms

   The coefficients (β) and the error terms (ϵ) are linear. The variables themselves might be linear or polynomials. But the coefficients and error terms must be linear.

2. The error term (ϵ) has population mean zero.

   In particular, the error term is normally distributed with the population mean of zero. If the population mean of the error term is other than zero, it means at least some part of the error term is predictable. Such a predictable part should be kept in the regression equation itself, not in error term. The error term should contain only random error, which can be attributed to chance. When the population mean is not equal to zero, statisticians call it “bias”.

3. None of the independent variables is correlated with the error term.

If any of the independent variable is correlated with the error term, it means it is possible to predict the error term. So, in such case, it is better to put this information in the regression equation itself. As we said in the previous point, the error term should contain only the random error, which explains the variability of the dependent variable. If the error term also explains the variability of any of the independent variable, it is not an error per se.

4. Observations in the error term are independent of one another (i.e., not correlated with one another). If error terms are correlated, it means it is possible to predict the error term of the next observation, using the information of the error term in the first observation. And any information, which can be used to predict the error term, should be put in the regression equation itself. The error terms should contain only the random component.

5. The variance in the error terms remain constant for all the observations. This assumption is also called homoskedasticity. PENDING

6. There is no perfect correlation between any of the independent variable with the dependent variable. If there is a perfect correlation between any two variables, then one of the two variables is unnecessary.

The Five Assumptions of Multiple Linear Regression - Statology

Model Building Strategies

also see pdf on steps to follow in multiple regression model building

13 Multiple Regression and Model Building | Quantitative Research Methods for Political Science, Public Policy and Public Administration: 4th Edition With Applications in R (bookdown.org)

Model Diagnostics

Chapter 13 Multiple Regression Models | Introduction to Statistics and Data Science (nps.edu)

Diagnostics_for_multiple_regression (stanford.edu)

Interpretation of Results of Multiple Linear Regression

Multicollinearity and Variable Selection

Model Selection Strategies

6.

Regression Analysis with Dummy Variables

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Incorporating categorical variables in regression analysis*
- *Creating and interpreting dummy variables in R*
- *Dummy variable pitfalls and remedies in econometric modeling*
- *Examples and case studies of dummy variable regression in economics using R*

- Incorporating categorical variables in regression analysis

- Creating and interpreting dummy variables in R
- Dummy variable pitfalls and remedies in econometric modeling
- Examples and case studies of dummy variable regression in economics using R

7.

Heteroscedasticity and Robust Regression

Status
This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Understanding heteroscedasticity and its implications*
- *Addressing heteroscedasticity using robust regression techniques in R*
- *Interpreting robust regression results in an economic context*
- *Practical examples and exercises showcasing robust regression in econometrics*

- Understanding heteroscedasticity and its implications

- Addressing heteroscedasticity using robust regression techniques in R
- Interpreting robust regression results in an economic context
- Practical examples and exercises showcasing robust regression in econometrics

8.

Time Series Regression

Status
This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Introduction to time series data in econometrics*
- *Time series regression models in R for economic analysis*
- *Dealing with autocorrelation and lagged variables*
- *Forecasting with time series regression models in R*
- *Applications of time series regression in economic forecasting*

- Introduction to time series data in econometrics

- Time series regression models in R for economic analysis
- Dealing with autocorrelation and lagged variables
- Forecasting with time series regression models in R
- Applications of time series regression in economic forecasting

9.

Introduction to Logistic Regression

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Basics of logistic regression in econometrics*
- *Estimating logistic regression models in R*
- *Interpreting logistic regression coefficients and odds ratios*
- *Applications of logistic regression in economic research using R*

- Basics of logistic regression in econometrics

- Estimating logistic regression models in R
- Interpreting logistic regression coefficients and odds ratios
- Applications of logistic regression in economic research using R

10

Model Evaluation and Selection

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Evaluating model performance and goodness-of-fit measures in econometrics*
- *Validation techniques for econometric regression models*
- *Comparing and selecting models using information criteria*
- *Cross-validation and bootstrapping for robust model assessment in econometrics*

- Evaluating model performance and goodness-of-fit measures in econometrics
- Validation techniques for econometric regression models
- Comparing and selecting models using information criteria

- Cross-validation and bootstrapping for robust model assessment in econometrics

11

Practical Tips and Resources for Econometric Regression

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Data preparation and preprocessing tips for econometric analysis*
- *Handling missing data and outliers in regression analysis*
- *Dealing with endogeneity and instrumental variables*
- *Additional resources for further learning and practice in econometrics with R*

- Data preparation and preprocessing tips for econometric analysis

- Handling missing data and outliers in regression analysis
- Dealing with endogeneity and instrumental variables
- Additional resources for further learning and practice in econometrics with R

12

Conclusion

Status

This chapter is currently a dumping ground for ideas, and we don't recommend reading it.

Learning Objectives:

In this chapter, you will learn to

- *Summary of the key concepts covered in the book*
- *Importance of regression analysis in econometrics and economic research*
- *Encouragement for further exploration and application of econometric regression using R*

- Summary of the key concepts covered in the book
- Importance of regression analysis in econometrics and economic research
- Encouragement for further exploration and application of econometric regression using R

R packages for econometric regression analysis and additional resources

This book could never be completed without using many packages. The most notable of the packages include tidyverse, easystats, AER, lmtest, fpp3, gujarati5sie, etc. These packages can be installed using the following commands in R.

```
RA_packages <- c("AER",
                 "easystats",
                 "fpp3",
                 "lmtest",
                 "ggthemes",
```

```
                    "gt",
                    "gtsummary",
                    "patchwork",
                    "here","fs",
                    "knitr",
                    "kableExtra")
install.packages(RA_packages)
```

The commands above install the packages into your R system. However, the functionality of these packages are not added into your R session just because you installed these packages. You have to load the required packages specifically whenever you need them.

Imagine you are building a home. You have completed electrification in your home. But just because you have a fan or an AC in your home doesn't mean they start automatically. You have to switch on the appliance whenever you need it. Similarly, you will have to load the R packages into your session whenever you need those packages. You can load the package using the `library(package_name)` command. For example, the command to load `easystats` set of packages would be

```
library(easystats)
```

When you run the `install.packages(package_name)` command, R installs the package from Comprehensive R Archive Network (CRAN), which is the highest authority to decide about R. However, there are many packages, which are not available on CRAN. These packages can be downloaded from code sharing platform Github.

To install packages from Github, you need to install one of the three packages first: `remotes`, `devtools` or `pak`. After that, you can easily install packages from Github also. Apart from such

packages, the development versions of regular packages (which are available on CRAN) can also be downloaded from Github. For example, if you want to install the package "gujarati5sie", which is a package containing data from the book "Basic Econometrics" written by Damodar Gujarati and others, then you can do it as follows:

```
#Install devtools, remotes or pak
install.packages("remotes")#or install.packages("devltools")
#or install.packages("pak")
#Load the package
library(remotes)#or library(devtools)
#or library(pak)
#download the gujarati5sie package
#using one of the above packages
remotes::install_github("bhattmaulik/Gujarati5sie")
#or devtools::install_github("bhattmaulik/Gujarati5sie")
#or pak::pak("bhattmaulik/Gujarati5sie")
```

B.

Data sets used in the book's examples

B.0.1. Introduction

Dummy text

C.

How to Build This Book Locally

C.0.1. Introduction

If you want to build this book locally on your computer, please download the entire code from Github. The first step is to visit the website `www.github.com/bhattmaulik/RegressionAnalysis`. Here, you will find the option to download the entire book in a zip file, as shown below.

```
knitr::include_graphics(here::here("images","download-book.png"))
```

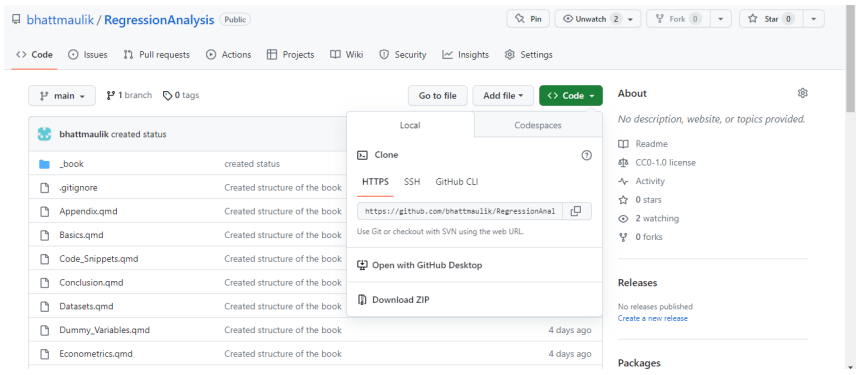

Figure C.1.: Download the book from Github

After you download the zip file, unzip it. This will create a folder in your computer. Within this folder, double click on the "RegressionAnalysis.Rproj" file. This will open the whole project in RStudio.

After opening this project, go to "Build" pane in RStudio. This pane is generally on the top right of RStudio along with Environment, History, Connections, Git and Tutorial. In the build pane, click on "`Render Book`" and select "`HTML format`".

You can also choose to build the book in PDF format, but for that you will need additional software called `tinytex`. In order to build a book in PDF format, R also needs `LaTeX` compiler. There are two popular `LaTeX` compilers: TeX Live and MikTeX. However, they have their own set of problems for R users. The `LaTeX` compiler `tinytex` attempts to solve many of them.

If you want to build the PDF book, you can install `tinytex` through terminal in RStudio using the command `quarto install tinytex`.

```
knitr::include_graphics(here::here("images","build-book.png"))
```

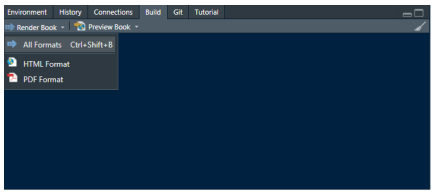

Figure C.2.: Build book using tinytex

If you can't install it this way, or if you want to install it through traditional way using R code, you can use the following code:

```
install.packages("tinytex")
library(tinytex)
tinytex::install_tinytex()
#if you want to install all the LaTeX packages,
#you can modify the command to
#tinytex::install_tinytex(bundle = "TinyTeX-2")
```

This confuses some readers because there are two `tinytex` in this code: the first tinytex is the `tinytex` R package. The other `tinytex` is the `LaTeX` compiler `tinytex`. So, when we write `library(tinytex)`, we are calling the R package tinytex. And when we use the command `install_tinytex`, we are installing the `LaTeX` compiler `tinytex` using the R package `tinytex`. The benefit of this approach is that you get to select which bundle you want to install. By default, you get to install the bundle `TinyTeX-1`, which contains only the most necessary LaTeX packages. But if you choose the bundle `TinyTeX-2`, you can download all the `LaTeX` packages.

References

Arnold, Jeffrey B. 2021. *ggthemes: Extra Themes, Scales and Geoms for "ggplot2"*. https://CRAN.R-project.org/package=ggthemes.

Hester, Jim, Hadley Wickham, and Gábor Csárdi. 2023. *fs: Cross-Platform File System Operations Based on "libuv"*. https://CRAN.R-project.org/package=fs.

Hyndman, Rob. 2023. *Fpp3: Data for "Forecasting: Principles and Practice" (3rd Edition)*. https://CRAN.R-project.org/package=fpp3.

Iannone, Richard, Joe Cheng, Barret Schloerke, Ellis Hughes, Alexandra Lauer, and JooYoung Seo. 2023. *gt: Easily Create Presentation-Ready Display Tables*. https://CRAN.R-project.org/package=gt.

Kleiber, Christian, and Achim Zeileis. 2008. *Applied Econometrics with R*. New York: Springer-Verlag. https://CRAN.R-project.org/package=AER.

Lüdecke, Daniel, Mattan S. Ben-Shachar, Indrajeet Patil, Brenton M. Wiernik, Etienne Bacher, Rémi Thériault, and Dominique Makowski. 2022. "easystats: Framework for Easy Statistical Modeling, Visualization, and Reporting." *CRAN*. https://easystats.github.io/easystats/.

Müller, Kirill. 2020. *here: A Simpler Way to Find Your Files*. https://CRAN.R-project.org/package=here.

Pedersen, Thomas Lin. 2022. *patchwork: The Composer of Plots*. https://CRAN.R-project.org/

package=patchwork.

Sjoberg, Daniel D., Karissa Whiting, Michael Curry, Jessica A. Lavery, and Joseph Larmarange. 2021. "Reproducible Summary Tables with the Gtsummary Package." *The R Journal* 13: 570–80. https://doi.org/10.32614/RJ-2021-053.

Xie, Yihui. 2014. "knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC.

———. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

———. 2023. *knitr: A General-Purpose Package for Dynamic Report Generation in r*. https://yihui.org/knitr/.

Zeileis, Achim, and Torsten Hothorn. 2002. "Diagnostic Checking in Regression Relationships." *R News* 2 (3): 7–10. https://CRAN.R-project.org/doc/Rnews/.

Zhu, Hao. 2021. *kableExtra: Construct Complex Table with "kable" and Pipe Syntax*. https://CRAN.R-project.org/package=kableExtra.