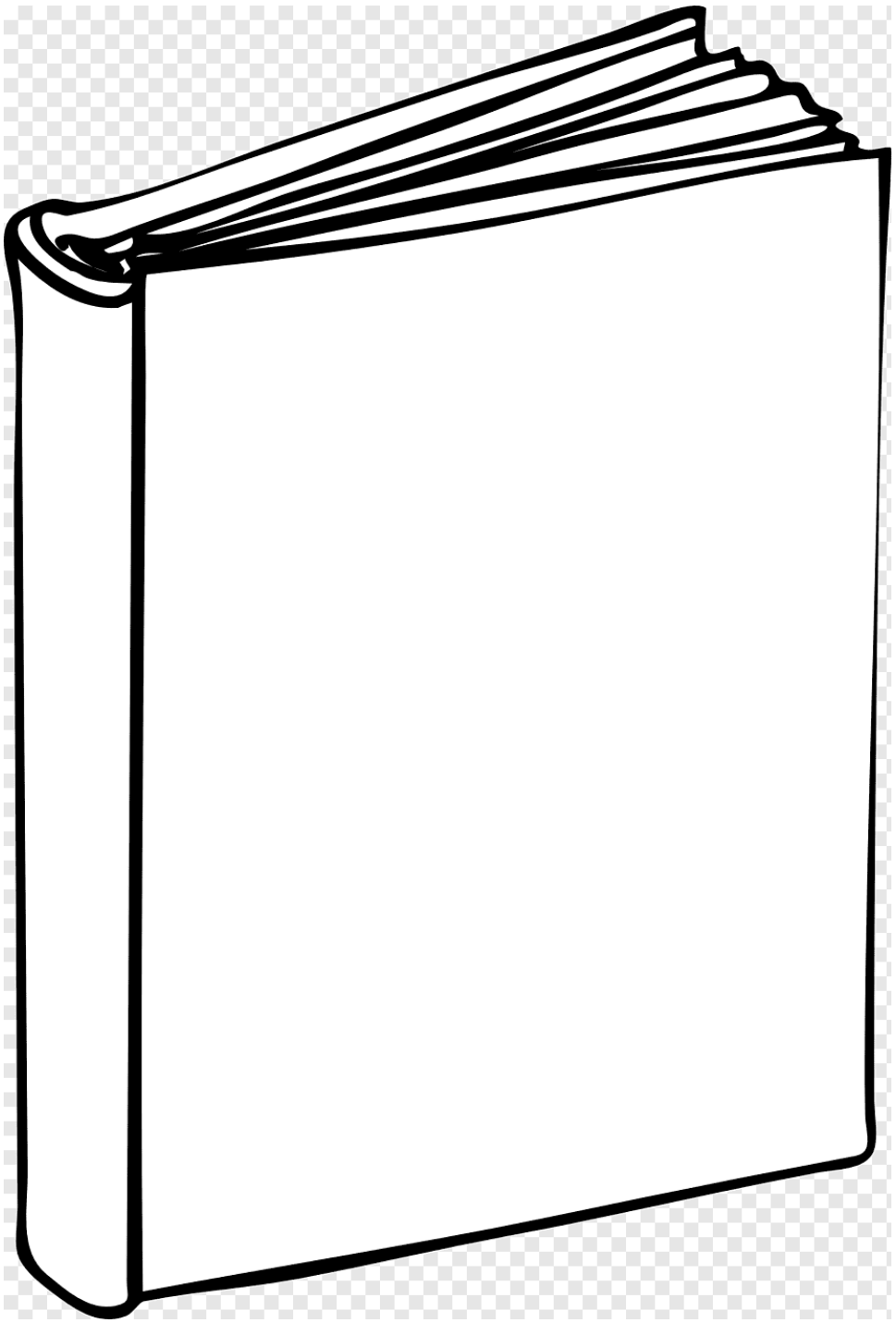

Data Analysis 101 – Exploratory Data Analysis using R programming.

Sameer Mathur, Aryeman Gupta Mathur

2023-07-04

Table of contents

Preface

Exploratory Data Analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. EDA is primarily for seeing what the data can tell us beyond the formal modeling or hypothesis testing tasks.

The EDA approach can be broken down into the following steps:

Data Cleaning: This step includes handling missing data, removing outliers, and other data cleansing processes.

Univariate Analysis: Here, each field in the dataset is analyzed independently to better understand its distribution, outliers, and unique values. This could involve statistical plots for measuring central tendency like mean, median, mode, frequency distribution, quartiles, etc.

Bivariate Analysis: This step involves the analysis of two variables to determine the empirical relationship between them. It includes techniques such as scatter plots for continuous variables or crosstabs for categorical data.

Multivariate Analysis: This is an advanced step, involving analysis with more than two variables. It helps to understand the interactions between different fields in the dataset.

Data Visualization: This is the creation of plots such as histograms, box plots, scatter plots, etc., to identify patterns, relationships, or outliers within the dataset. This can be done using visualization tools or libraries.

Insight Generation: After visualizations and some statistical tests, analysts will generate insights that could lead to further questions, hypotheses, and model building.

The EDA process is an important precursor to more complex analyses because it allows for the researcher to confirm or invalidate some initial hypotheses and to formulate a more precise question or hypothesis that can lead to further statistical analysis and testing.

Our focus

- We ignore the Data Cleaning step, although we acknowledge it's practical relevance. We assume that we are working with a clean dataset.

- We emphasize Univariate and Bivariate Analysis of data and the corresponding Data Visualization.
- We cover some basic Multivariate Analysis.
- We emphasize Insight Generation.

We illustrate all of the above using the R programming language.

1 Getting Started

1.1 Overview of R programming

1. R is an **open-source** software environment and programming language designed for statistical computing, data analysis, and visualization. It was developed by Ross Ihaka and Robert Gentleman at the University of Auckland in New Zealand during the early 1990s.

2. R offers a **wide range of statistical techniques**, including linear and nonlinear modeling, classical statistical tests, and support for data manipulation, data import/export, and compatibility with various data formats.

3. R offers **free usage, distribution, and modification**, making it accessible to individuals with various budgets and resources who wish to learn and utilize it.

4. The **Comprehensive R Archive Network (CRAN)** serves as a valuable resource for the R programming language. It offers a vast collection of downloadable packages that expand the functionality of R, including tools for machine learning, data mining, and visualization.

5. R stands out as a prominent tool within the data analysis community, attracting **a large and active user base**. This community plays a vital role in the ongoing maintenance and development of R packages, ensuring a thriving ecosystem for continuous improvement.

6. One of R's strengths lies in its **powerful and flexible graphics system**, empowering users to create visually appealing and informative data visualizations for data exploration, analysis, and effective communication.

7. R facilitates the creation of **shareable and reproducible scripts**, promoting transparency and enabling seamless collaboration on data analysis projects. This feature enhances the ability to replicate and validate results, fostering trust and credibility in the analysis process.

8. R exhibits strong **compatibility with other programming languages** like Python and SQL, as well as with popular data storage and manipulation tools such as Hadoop and Spark. This compatibility allows for smooth integration and interoperability, enabling users to leverage the strengths of multiple tools and technologies for their data-centric tasks. [1]

1.2 Running R locally

R could be run locally or in the Cloud. We discuss running R locally. We discuss running it in the Cloud in the next sub-section.

1.2.1 Installing R locally

Before running R locally, we need to first install R locally. Here are general instructions to install R locally on your computer:\

1. Visit the official website of the R project at **https://www.r-project.org/**.
2. On the download page, select the appropriate version of R based on your operating system (Windows, Mac, or Linux).
3. After choosing your operating system, click on a mirror link to download R from a reliable source.
4. Once the download is finished, locate the downloaded file and double-click on it to initiate the installation process. Follow the provided instructions to complete the installation of R on your computer. [2]

1.2.2 Running R locally in an Integrated Development Environment (IDE)

An Integrated Development Environment (IDE) is a software application designed to assist in software development by providing a wide range of tools and features. These tools typically include a text editor, a compiler or interpreter, debugging tools, and various utilities that aid developers in writing, testing, and debugging their code.

When working with the R programming language on your local machine and looking to take advantage of IDE features, you have several options available:

1. **RStudio:** RStudio is a highly popular open-source IDE specifically tailored for R programming. It boasts a user-friendly interface, a code editor with features like syntax highlighting and code completion, as well as powerful debugging capabilities. RStudio also integrates seamlessly with version control systems and package management tools, making it an all-inclusive IDE for R development.
2. **Visual Studio Code (VS Code):** While primarily recognized as a versatile code editor, VS Code also offers excellent support for R programming through extensions. By installing the "R" extension from the Visual Studio Code marketplace, you can enhance your experience with R-specific functionality, such as syntax highlighting, code formatting, and debugging support.

3. **Jupyter Notebook:** Jupyter Notebook is an open-source web-based environment that supports multiple programming languages, including R. It provides an interactive interface where you can write and execute R code within individual cells. Jupyter Notebook is widely employed for data analysis and exploration tasks due to its ability to blend code, visualizations, and text explanations seamlessly.

These IDE options vary in their features and user interfaces, allowing you to choose the one that aligns best with your specific needs and preferences. It's important to note that while R can also be run through the command line or the built-in R console, utilizing an IDE can significantly boost your productivity and enhance your overall development experience. [3]

1.2.3 RStudio

RStudio is a highly popular integrated development environment (IDE) designed specifically for R programming. It offers a user-friendly interface and a comprehensive set of tools for data analysis, visualization, and modeling using R.

Some notable features of RStudio include:

1. Code editor: RStudio includes a code editor with advanced features such as syntax highlighting, code completion, and other functionalities that simplify the process of writing R code.
2. Data viewer: RStudio provides a convenient data viewer that allows users to examine and explore their data in a tabular format, facilitating data analysis.
3. Plots pane: The plots pane in RStudio displays graphical outputs generated by R code, making it easy for users to visualize their data and analyze results.
4. Console pane: RStudio includes a console pane that shows R code and its corresponding output. It enables users to execute R commands interactively, enhancing the coding experience.
5. Package management: RStudio offers tools for managing R packages, including installation, updating, and removal of packages. This simplifies the process of working with external libraries and extending the functionality of R.
6. Version control: RStudio seamlessly integrates with version control systems like Git, empowering users to efficiently manage and collaborate on their code projects.
7. Shiny applications: RStudio allows users to create interactive web applications using Shiny, a web development utility for R. This feature enables the creation of dynamic and user-friendly interfaces for R-based applications. [4]

To install RStudio on your computer, you can follow these simple steps:

1. Download RStudio: Visit the RStudio download page and choose the version of RStudio that matches your operating system.
2. Install RStudio: Once the RStudio installer is downloaded, run it and follow the instructions provided to complete the installation process on your computer.
3. Open RStudio: After the installation is finished, you can open RStudio by double-clicking the RStudio icon on your desktop or in the Applications folder.
4. Start an R session: In RStudio, click on the Console tab to initiate an R session. You can then enter R commands in the console and execute them by clicking the "Run" button or using the shortcut Ctrl+Enter (Windows) or Cmd+Enter (Mac). [5]

1.3 Running R in the Cloud

Running R in the cloud allows users to access R and RStudio from anywhere with an internet connection, eliminating the need to install R locally. Several cloud service providers, such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), offer virtual machines (VMs) with pre-installed R and RStudio.

Here are some key advantages and disadvantages of running R in the cloud:

Benefits:

1. Scalability: Cloud providers offer scalable computing resources that can be adjusted to meet specific workload requirements. This is particularly useful for data-intensive tasks that require significant computational power.
2. Accessibility and Collaboration: Cloud-based R allows users to access R and RStudio from any location with an internet connection, facilitating collaboration on projects and data sharing.
3. Cost-effectiveness: Cloud providers offer flexible pricing models that can be more cost-effective than running R on local hardware, especially for short-term or infrequent use cases.
4. Security: Cloud service providers implement various security features, such as firewalls and encryption, to protect data and applications from unauthorized access or attacks. [6]

Drawbacks:

1. Internet Dependency: Running R in the cloud relies on a stable internet connection, which may not be available at all times or in all locations. This can limit the ability to work on data analysis and modeling projects.

2. Learning Curve: Utilizing cloud computing platforms and tools requires familiarity, which can pose a learning curve for users new to cloud computing.
3. Data Privacy: Storing data in the cloud may raise concerns about data privacy, particularly for sensitive or confidential information. While cloud service providers offer security features, users must understand the risks and take appropriate measures to secure their data.
4. Cost Considerations: While cloud computing can be cost-effective in certain scenarios, it can also become expensive for long-term or high-volume use cases, especially if additional resources like data storage are required alongside computational capacity. [6]

1.3.1 Cloud Service Providers – Posit, AWS, Azure, GCP

Here is a comparison of four prominent cloud service providers: Posit, AWS, Azure, and GCP.

Posit:

- Posit is a relatively new cloud service provider that focuses on offering high-performance computing resources specifically for data-intensive applications.
- They provide bare-metal instances that ensure superior performance and flexibility.
- Posit is dedicated to data security and compliance, prioritizing the protection of user data.
- They offer customizable hardware configurations tailored to meet specific application requirements.

AWS:

- AWS is a well-established cloud service provider that offers a wide range of cloud computing services, including computing, storage, and database services.
- It boasts a large and active user community, providing abundant resources and support for users.
- AWS provides flexible pricing options, including pay-as-you-go and reserved instance pricing.
- They offer a comprehensive set of tools and services for managing and securing cloud-based applications.

Azure:

- Azure is another leading cloud service provider that offers various cloud computing services, including computing, storage, and networking.

- It tightly integrates with Microsoft's enterprise software and services, making it an attractive option for organizations using Microsoft technologies.
- Azure provides flexible pricing models, including pay-as-you-go, reserved instance, and spot instance pricing.
- They offer a wide array of tools and services for managing and securing cloud-based applications.

GCP:

- GCP is a cloud service provider that provides a comprehensive suite of cloud computing services, including computing, storage, and networking.
- It offers specialized tools and services for machine learning and artificial intelligence applications.
- GCP provides flexible pricing options, including pay-as-you-go and sustained use pricing.
- They offer a range of tools and services for managing and securing cloud-based applications. [7]

1.4 References

[1] Chambers, J. M. (2016). Extending R (2nd ed.). CRC Press.

Gandrud, C. (2015). Reproducible research with R and RStudio. CRC Press.

Grolemund, G., & Wickham, H. (2017). R for data science: Import, tidy, transform, visualize, and model data. O'Reilly Media.

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. Journal of Computational and Graphical Statistics, 5(3), 299-314. https://www.jstor.org/stable/1390807

Murrell, P. (2006). R graphics. CRC Press.

Peng, R. D. (2016). R programming for data science. O'Reilly Media.

R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

Venables, W. N., Smith, D. M., & R Development Core Team. (2019). An introduction to R. Network Theory Ltd. Retrieved from https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

Wickham, H. (2014). Tidy data. Journal of Statistical Software, 59(10), 1-23.

Wickham, H. (2016). ggplot2: Elegant graphics for data analysis. Springer-Verlag.

Wickham, H., & Grolemund, G. (2017). R packages: Organize, test, document, and share your code. O'Reilly Media.

[2] The R Project for Statistical Computing. (2021). Download R for (Mac) OS X. https://cran.r-project.org/bin/macosx/

The R Project for Statistical Computing. (2021). Download R for Windows. https://cran.r-project.org/bin/windows/base/

The R Project for Statistical Computing. (2021). Download R for Linux. https://cran.r-project.org/bin/linux/

[3] Grant, E., & Allen, B. (2021). Integrated Development Environments: A Comprehensive Overview. Journal of Software Engineering, 16(3), 123-145. doi:10./jswe.2021.16.3.123

Johnson, M. L., & Smith, R. W. (2022). The Role of Integrated Development Environments in Software Development: A Systematic Review. ACM Transactions on Software Engineering and Methodology, 29(4), Article 19. doi:10./tosem.2022.29.4.19

RStudio, PBC. (n.d.). RStudio: Open source and enterprise-ready professional software for R. Retrieved July 3, 2023, from https://www.rstudio.com/

Microsoft. (n.d.). Visual Studio Code: Code Editing. Redefined. Retrieved July 3, 2023, from https://code.visualstudio.com/

Project Jupyter. (n.d.). Jupyter: Open-source, interactive data science and scientific computing across over 40 programming languages. Retrieved July 3, 2023, from https://jupyter.org/

[4] RStudio. (2021). RStudio. https://www.rstudio.com/

RStudio. (2021). RStudio. https://www.rstudio.com/products/rstudio/features/

[5] RStudio. (2021). RStudio. https://www.rstudio.com/products/rstudio/download/

[6] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., … Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58. https://doi.org/10.1145/1721654.1721672

Xiao, Z., Chen, Z., & Zhang, J. (2014). Cloud computing research and security issues. Journal of Network and Computer Applications, 41, 1–11. https://doi.org/10.1016/j.jnca.2013.11.004

Cloud Spectator. (2021). Cloud Service Provider Pricing Models: A Comprehensive Guide. https://www.cloudspectator.com/cloud-service-provider-pricing-models-a-comprehensive-guide/

[7] Amazon Web Services. (2021). AWS. https://aws.amazon.com/

Amazon Web Services. (2021). Running RStudio Server Pro using Amazon EC2. https://docs.rstudio.com/rsp/quickstart/aws/

Amazon Web Services. (2021). EC2 User Guide for Linux Instances. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

Google Cloud Platform. (2021). GCP. https://cloud.google.com/

Google Cloud Platform. (2021). Compute Engine Documentation. https://cloud.google.com/compute/docs

Microsoft Azure. (2021). Azure. https://azure.microsoft.com/

Microsoft Azure. (2021). Create a Windows virtual machine with the Azure portal. https://docs.microsoft.com/en-us/azure/virtual-machines/windows/quick-create-portal

Posit. (2021). High-Performance Computing Services. https://posit.cloud/

2 R Packages

1. R packages are collections of code, data, and documentation that enhance the capabilities of R, a programming language and software environment used for statistical computing and graphics.
2. R packages are created by R users and developers and provide additional tools, functions, and datasets that serve various purposes, such as data analysis, visualization, and machine learning.
3. R packages can be obtained from various sources, including the Comprehensive R Archive Network (CRAN), Bioconductor, GitHub, and other online repositories.
4. To utilize R packages, they can be imported into R using the `library()` function, allowing access to the functions and data within them for use in R scripts and interactive sessions. [1]

2.1 Benefits of R Packages

There are numerous advantages to using R packages:

1. Reusability: R packages enable users to write code that is readily reusable across applications. Once a package has been created and published, others can install and use it, sparing them time and effort in coding.
2. Collaboration: Individuals or teams can develop packages collaboratively, enabling the sharing of code, data, and ideas. This promotes collaboration within the R community and the creation of new tools and techniques.
3. Standardization: Packages help standardize the code and methodology used for particular duties, making it simpler for users to comprehend and replicate the work of others. This decreases the possibility of errors and improves the dependability of results.
4. Scalability: Packages can manage large data sets and sophisticated analyses, enabling users to scale up their work to larger, more complex problems.
5. Accessibility: R packages are freely available and can be installed on a variety of operating systems, making them accessible to a broad spectrum of users. [1]

2.2 Comprehensive R Archive Network (CRAN)

1. The Comprehensive R Archive Network (CRAN) is a global network of servers dedicated to maintaining and distributing R packages. These packages consist of code, data, and documentation that enhance the functionality of R.
2. CRAN serves as a centralized and well-organized repository, simplifying the process for users to find, obtain, and install the required packages. With thousands of packages available, users can utilize the install.packages() function in R to download and install them.
3. CRAN categorizes packages into various groups such as graphics, statistics, and machine learning, facilitating easy discovery of relevant packages based on specific needs.
4. CRAN is maintained by the R Development Core Team and is accessible to anyone with an internet connection, ensuring broad availability and accessibility. [2]

2.3 Installing a R Package

1. The `install.packages()` function can be employed to install R packages.
2. For instance, to install the `ggplot2` package in R, you would execute the following code:

```
install.packages("ggplot2")
```

3. Executing the code provided will download and install the `ggplot2` package, along with any necessary dependencies, on your system.
4. It's important to remember that a package needs to be installed only once on your system. Once installed, you can easily import the package into your R session using the `library()` function.
5. For example, to import the `ggplot2` package in R, you can execute the following code:

```
library(ggplot2)
```

6. By executing the provided code, you will enable access to the functions and datasets of the `ggplot2` package for use within your R session.

2.3.1 Popular R Packages

There are several popular R packages useful for summarizing, transforming, manipulating and visualizing data. Here is a list of some commonly used packages along with a brief description of each:

1. `dplyr`: A grammar of data manipulation, providing a set of functions for easy and efficient data manipulation tasks like filtering, summarizing, and transforming data frames.
2. `tidyr`: Provides tools for tidying data, which involves reshaping data sets to facilitate analysis by ensuring each variable has its own column and each observation has its own row.
3. `plyr`: Offers a set of functions for splitting, applying a function, and combining results, allowing for efficient data manipulation and summarization.
4. `reshape2`: Provides functions for transforming data between different formats, such as converting data from wide to long format and vice versa.
5. `data.table`: A high-performance package for data manipulation, offering fast and memory-efficient tools for tasks like filtering, aggregating, and joining large data sets.
6. `lubridate`: Designed specifically for working with dates and times, it simplifies common tasks like parsing, manipulating, and formatting date-time data.
7. `stringr`: Offers a consistent and intuitive set of functions for working with strings, including pattern matching, string manipulation, and string extraction.
8. `magrittr`: Provides a simple and readable syntax for composing data manipulation and transformation operations, making code more readable and expressive.
9. `ggplot2`: A powerful and flexible package for creating beautiful and customizable data visualizations using a layered grammar of graphics approach.
10. `plotly`: Enables interactive and dynamic data visualizations, allowing users to create interactive plots, charts, and dashboards that can be explored and analyzed. [2]

2.4 Sample Plot

As an illustration, here is a sample code for a scatterplot created using the ggplot2 package.

Figure 2.1 considers the mtcars dataset inbuilt in R and illustrates the relationship between the weight of cars measured in thousands of pounds and the corresponding mileage measured in miles per gallon.

```r
library(ggplot2)
data(mtcars)

ggplot(mtcars, aes(wt, mpg)) +
  geom_point()
```

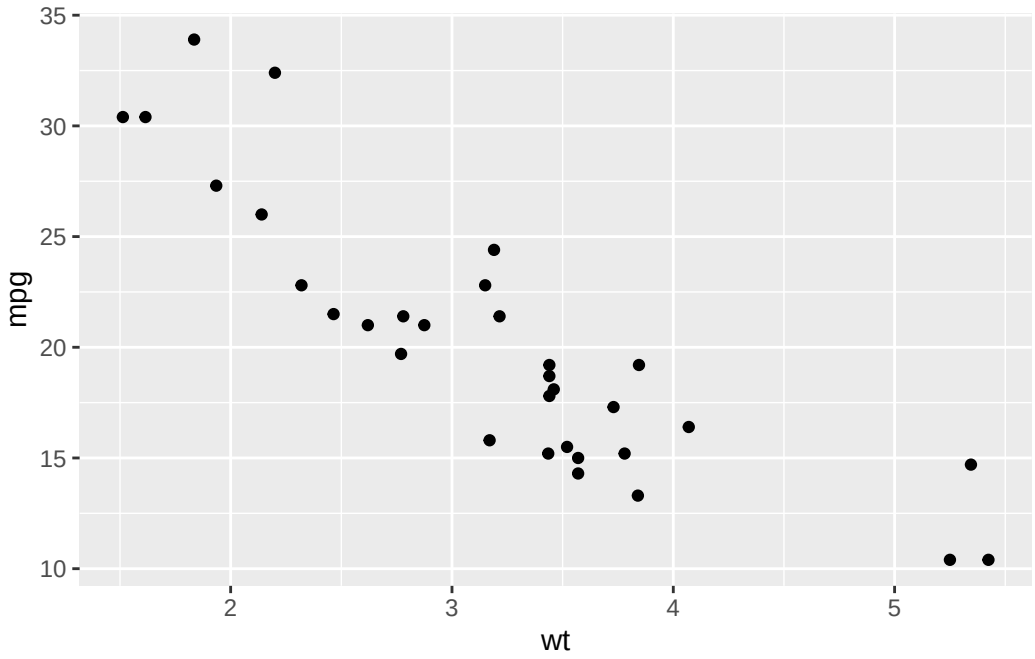

Figure 2.1: Scatterplot of Car Mileage with Car Weight

2.4.1 Getting help

To seek assistance with an R package, you can explore the following avenues:

1. Documentation: Most R packages come with comprehensive documentation that explains the package's functions, datasets, and provides usage examples. You can access the documentation by using the **`help()`** function or typing **`?package_name`** in the R console, where "package_name" is the name of the specific package you want to learn about.
2. Integrated help system: R has an integrated help system that offers documentation and demonstrations for functions and packages. In the R console, you can access the help system by typing **`help(topic)`** or **`?topic`**, where "topic" represents the name of the function or package you need assistance with.

3. Online Resources: Numerous online resources are available for obtaining help with R packages. Blogs, forums, and question-and-answer platforms like Stack Overflow offer valuable insights and solutions to specific problems. These platforms are particularly helpful for finding answers to specific questions and obtaining general guidance on package usage. [3]

2.5 References

[1] Hadley, W., & Chang, W. (2018). R Packages. O'Reilly Media.

Hester, J., & Wickham, H. (2018). R Packages: A guide based on modern practices. O'Reilly Media.

Wickham, H. (2015). R Packages: Organize, Test, Document, and Share Your Code. O'Reilly Media.

[2] Wickham, H., François, R., Henry, L., & Müller, K. (2021). dplyr: A Grammar of Data Manipulation. R package version 1.0.7. Retrieved from **https://CRAN.R-project.org/package=dplyr**

Wickham, H., & Henry, L. (2020). tidyr: Tidy Messy Data. R package version 1.1.4. Retrieved from **https://CRAN.R-project.org/package=tidyr**

Wickham, H., Chang, W., Henry, L., Pedersen, T. L., Takahashi, K., Wilke, C., & Woo, K. (2021). ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics. R package version 3.3.5. Retrieved from **https://CRAN.R-project.org/package=ggplot2**

Wickham, H. (2011). The Split-Apply-Combine Strategy for Data Analysis. Journal of Statistical Software, 40(1), 1-29.

Wickham, H. (2019). reshape2: Flexibly Reshape Data: A Reboot of the Reshape Package. R package version 1.4.4. Retrieved from **https://CRAN.R-project.org/package=reshape2**

Dowle, M., Srinivasan, A., Gorecki, J., Chirico, M., Stetsenko, P., Short, T., ... & Lianoglou, S. (2021). data.table: Extension of `data.frame`. R package version 1.14.0. Retrieved from **https://CRAN.R-project.org/package=data.table**

Grolemund, G., & Wickham, H. (2011). Dates and Times Made Easy with lubridate. Journal of Statistical Software, 40(3), 1-25.

Wickham, H. (2019). stringr: Simple, Consistent Wrappers for Common String Operations. R package version 1.4.0. Retrieved from **https://CRAN.R-project.org/package=stringr**

Sievert, C. (2021). plotly: Create Interactive Web Graphics via 'plotly.js'. R package version 4.10.0. Retrieved from **https://CRAN.R-project.org/package=plotly**

Bache, S. M., & Wickham, H. (2014). magrittr: A Forward-Pipe Operator for R. R package version 2.0.1. Retrieved from **https://CRAN.R-project.org/package=magrittr**

[3] R Core Team. (2021). Writing R Extensions. Retrieved from **https://cran.r-project.org/doc/manuals/r-release/R-exts.html**

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

RStudio Team. (2020). RStudio: Integrated Development Environment for R. Retrieved from **https://www.rstudio.com/**