How to Create a Pareto Chart in R (Step-by-Step)

Authored by

stats writer

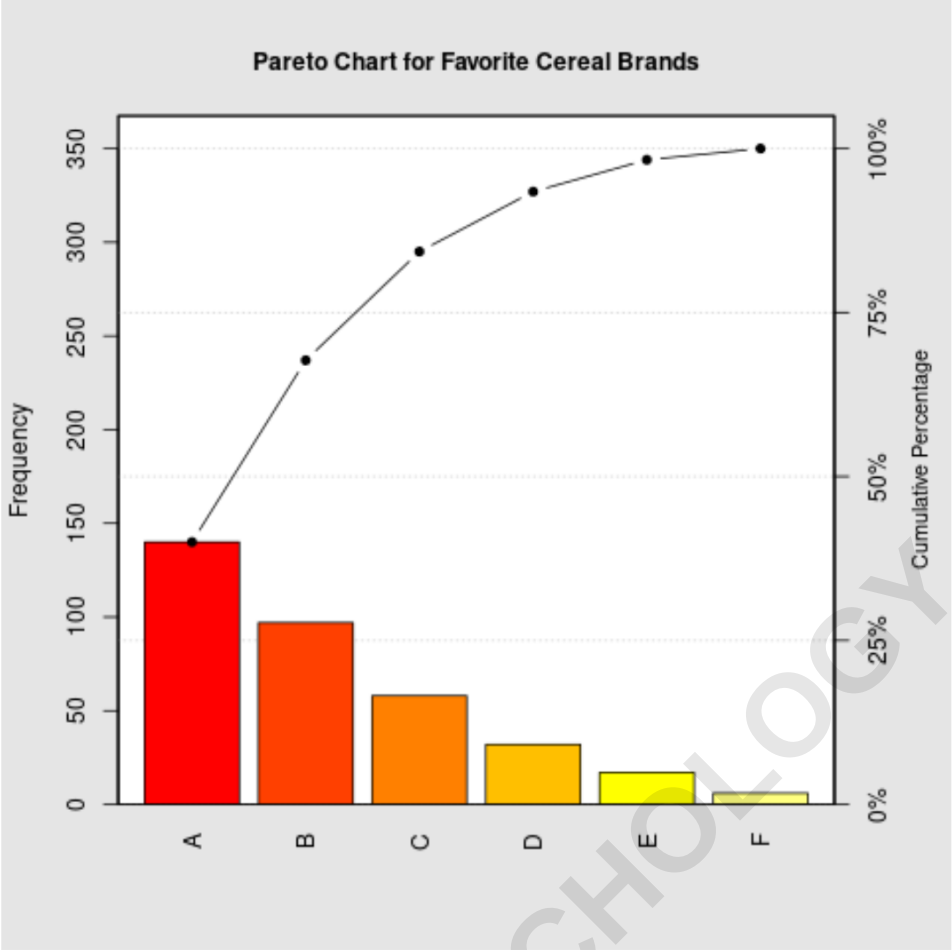
December 9, 2025

A Pareto chart is an essential tool in statistical analysis and quality control, designed to visually represent the most significant factors contributing to a specific outcome. This powerful visualization combines a bar chart, showing individual factor frequencies, and a line graph, displaying cumulative percentages, offering immediate insight into prioritization.

To successfully generate and customize this specific visualization within the R programming language, we will leverage specialized packages built for statistical quality control. Unlike standard plotting methods that might require extensive manual calculations for cumulative frequencies, utilizing the focused tools within packages like qcc package streamlines the process dramatically. This guide details the step-by-step process for creating a robust and informative Pareto chart, ensuring you can effectively analyze your frequency data.

The resulting chart illustrates the principle that 80% of problems often stem from 20% of causes--a concept rooted in the Pareto Principle. Mastering this technique in R allows analysts, engineers, and data scientists to quickly identify and focus on the categories that yield the highest impact, leading to more efficient resource allocation and problem-solving strategies.

A **Pareto chart** is a fundamental visualization tool that displays the frequencies of different categories, ordered from greatest to least, alongside the corresponding cumulative frequencies.



This comprehensive tutorial provides a clear, step-by-step example of how to create a statistically accurate Pareto chart using the powerful capabilities of the R programming language and the dedicated **qcc** package.

Prerequisites: Installing the Necessary Package

Before initiating the data analysis workflow, it is crucial to ensure that the required statistical packages are installed and loaded into your R session. For generating the Pareto chart, we rely specifically on the **qcc package**, which stands for "Quality Control Charts." This package is optimized for various quality control visualizations and provides the dedicated function we need.

If you have not installed this package previously, you must execute the installation command within your R console. Once installed, the library must be loaded for its functions to be accessible during the current session. These initial steps guarantee that the core functionality required for chart generation is available, preventing errors in the subsequent code execution.

The installation and loading process is straightforward, ensuring that even users new to R

programming language can quickly prepare their environment. The simplicity of the **qcc package** makes it ideal for quality control visualization tasks where speed and clarity are paramount.

Step 1: Preparing the Sample Data

The foundation of any statistical visualization is clean and well-structured data. For a Pareto analysis, the input data must represent categories and their corresponding frequencies or counts. The categories represent the factors or causes being analyzed, and the counts represent how often they occur or their magnitude of contribution.

For this example, we will model a hypothetical scenario derived from a market survey. Suppose a survey was conducted where 350 different individuals were asked to identify their favorite cereal brand among six options, labeled A through F. The counts collected represent the total votes for each brand, providing the necessary frequency distribution for our analysis.

In R, we structure this frequency data into a **data frame**, which is the standard structure for handling tabular data. We define two vectors: one for the categorical factors (cereal brands) and one for the numerical counts (votes). This structured approach ensures the data is easily consumable by the specialized charting functions in the **qcc** package.

The following dataset demonstrates the creation of this structure, defining the favorite brands and their respective vote totals:

#create data
df <- data.frame(favorite=c('A', 'B', 'C', 'D', 'E', 'F'),
count=c(140, 97, 58, 32, 17, 6))

#view data
df

favorite count
1 A 140
2 B 97
3 C 58
4 D 32
5 E 17
6 F 6

Step 2: Generating the Basic Pareto Chart

Once the data is prepared and the **qcc package** is loaded, generating the chart is exceptionally

straightforward. The package provides the dedicated function `pareto.chart()`, which handles all necessary internal calculations--such as sorting the categories, calculating cumulative frequencies, and generating the dual-axis plot--with a single command.

The primary input required by the `pareto.chart()` function is a numeric vector containing the frequencies or counts. In our structured data frame `df`, this corresponds specifically to the `df$count` column. The function automatically recognizes that these counts represent the observed frequencies of an underlying categorical variable, even though we only pass the numeric vector.

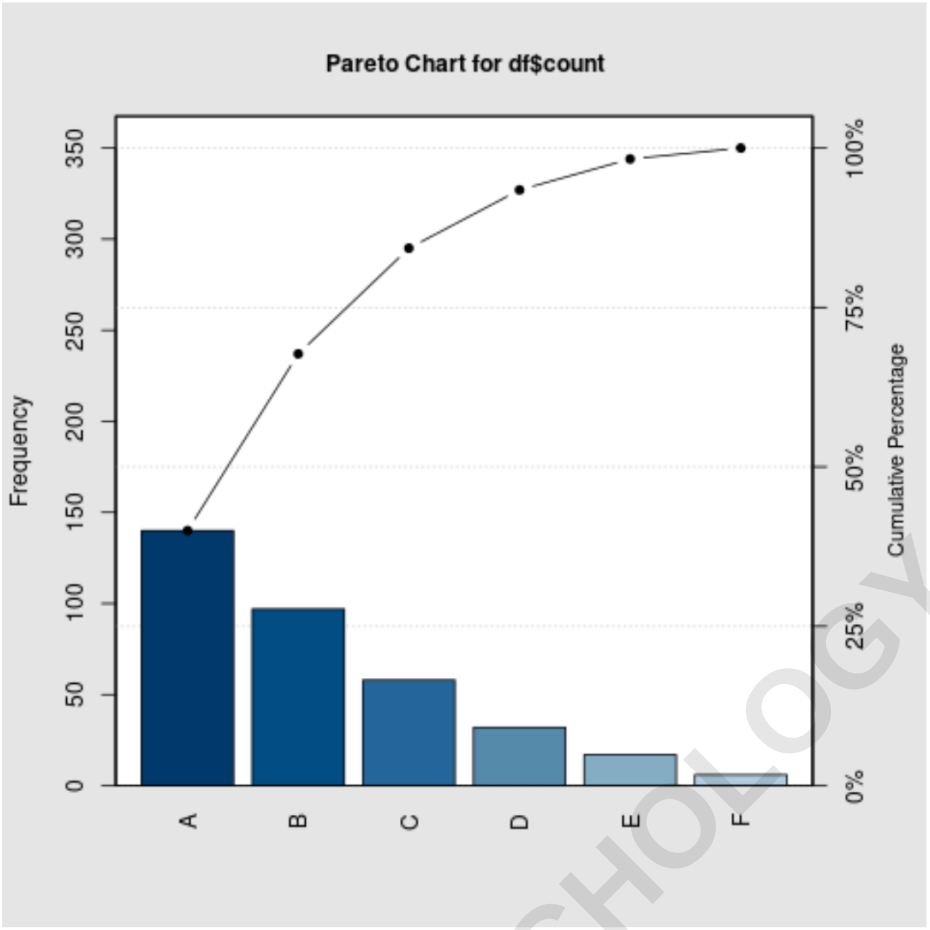
Upon execution, the function not only produces the visual chart output but also prints a detailed summary table directly to the console. This table is invaluable for confirming the calculations and understanding the precise numerical breakdown of the Pareto analysis. It clearly shows the frequency, cumulative frequency, percentage, and cumulative percentage for each category.

To generate the basic Pareto chart for our cereal brand survey, execute the following commands, ensuring you load the **qcc** library first:

library(qcc)

#create Pareto chart
pareto.chart(df$count)

Pareto chart analysis for df$count
Frequency Cum.Freq. Percentage Cum.Percent.
A 140.000000 140.000000 40.000000 40.000000
B 97.000000 237.000000 27.714286 67.714286
C 58.000000 295.000000 16.571429 84.285714
D 32.000000 327.000000 9.142857 93.428571
E 17.000000 344.000000 4.857143 98.285714
F 6.000000 350.000000 1.714286 100.000000



Interpreting the Output and Visualization

The visual output provided by `pareto.chart()` is a dual-axis plot. The vertical bars, read against the left y-axis, display the absolute frequencies of each category, sorted in descending order. This visual arrangement immediately highlights the most frequent causes or factors. The line graph, read against the right y-axis, tracks the cumulative percentage, which is the defining characteristic of this type of chart.

The summary table produced in the console offers a precise numerical confirmation of the visualization. This table is ordered identically to the chart, starting with the category that holds the highest frequency. Understanding these columns is key to applying the Pareto Principle, or the 80/20 rule, to the data.

The critical data points derived from the output allow us to draw specific conclusions regarding our survey results. For example, by examining the cumulative percentage column, we can determine how many brands account for 80% of the total votes. The data shows that Brand A accounts for 40% alone, and Brands A, B, and C combined reach 84.28% of the total votes. This suggests that

focusing resources on improving the offering or distribution of brands A, B, and C would address the vast majority of consumer preference captured in the survey.

A closer look at the calculated metrics for the first few categories reveals the compounding effect of the cumulative frequency:

Frequency of brand A: **140** | Cumulative frequency: **140** (40.00% of the total).
Frequency of brand B: **97** | Cumulative frequency of A, B: **237** (67.71% of the total).
Frequency of brand C: **58** | Cumulative frequency of A, B, C: **295** (84.28% of the total).

These metrics demonstrate clearly that Brands A, B, and C constitute the critical few, while D, E, and F represent the trivial many, highlighting where managerial attention should be concentrated. The remaining categories, D, E, and F, collectively contribute only about 15.72% of the total votes, illustrating the power of this statistical focus.

Step 3: Customizing the Appearance

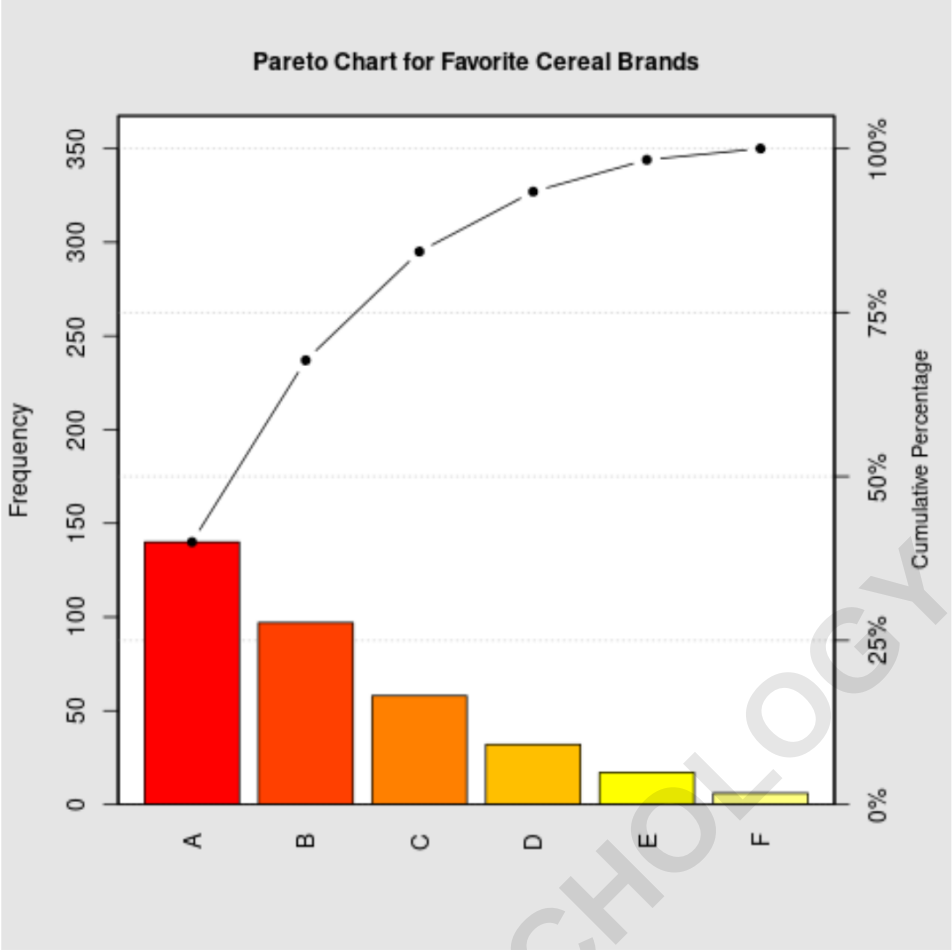
While the default output of the `pareto.chart()` function is functional and statistically accurate, customization is often necessary for professional presentation or clarity. The function allows users to modify several key graphical parameters directly through its arguments, aligning the visual output with specific report requirements or organizational branding.

Two of the most common and impactful modifications involve setting a descriptive title and defining a specific color palette. The `main` argument controls the overall title of the chart, providing essential context to the viewer. For color modification, the `col` argument accepts a vector of colors or, more practically, a function that generates a color palette, such as `heat.colors()`, `rainbow()`, or `topo.colors()`.

When selecting colors, it is important to ensure that the number of colors generated matches the number of categories being plotted. Using an R function like `length(df$count)` dynamically ensures that the palette contains the correct number of distinct colors, preventing potential errors and maximizing clarity in the visualization.

The following R code demonstrates how to apply a customized title and use the built-in `heat.colors()` function to generate a specific aesthetic for the bars in the R programming language plot:

```
pareto.chart(df$count,
main='Pareto Chart for Favorite Cereal Brands',
col=heat.colors(length(df$count)))
```



For advanced customization beyond simple colors and titles, users may explore the extensive documentation available for graphical parameters (par) in R, which allows fine-tuning elements like axis labels, font sizes, and margins. For instance, detailed control over the plot axes or specific text annotations would utilize standard R plotting functions in conjunction with the output of `pareto.chart()`.

Advanced Considerations and Data Structuring

While this guide focuses on using a pre-aggregated numeric vector (counts), it is vital to understand how to handle raw, unaggregated categorical data. If your input data consists of thousands of individual survey responses (e.g., a long list of brand names), you would first need to aggregate or tabulate these results to derive the required frequency counts. This pre-processing step is typically handled using functions like `table()` or data manipulation packages like **dplyr** within R.

Furthermore, in scenarios where the categories are not implicitly ordered (as is the case when passing a simple numeric vector), you may need to explicitly label the categories to ensure clarity

in the output chart. While `pareto.chart()` attempts to use indices if no names are provided, supplying a named vector or using factors correctly ensures the labels in the plot accurately reflect the items being measured.

For more complex visualizations, such as interactive plots or integration into dashboards, users might consider using popular visualization packages like **ggplot2**. Although **ggplot2** does not have a native, single function for the Pareto chart, users can calculate the necessary cumulative statistics manually and then plot the bars and lines separately, offering maximum control over every aesthetic detail, though requiring more coding effort than the specialized **qcc package** approach.

The choice between using the specialized **qcc** function and a general-purpose package like **ggplot2** generally depends on the goal: use **qcc** for quick, robust quality control charts where standardization is acceptable, and use **ggplot2** when absolute control over every visual element is required, especially when integrating the chart into a larger, multi-layered visual narrative. In either case, the initial step of transforming raw counts into a usable **data frame** remains constant.

Conclusion: Leveraging Pareto Analysis in R

The ability to generate a clean and insightful Pareto chart in R programming language using the `pareto.chart()` function from the **qcc** package is a valuable skill for any data analyst focused on efficiency and quality improvement. This visualization technique moves beyond simple frequency analysis by forcing attention onto the few vital contributors responsible for the majority of the observations.

By following the steps outlined--from meticulous data preparation into a functional **data frame** to applying the core charting function and performing optional customizations--you can quickly transform raw frequency data into actionable business intelligence. The resulting chart serves as a powerful communication tool, clearly justifying resource allocation and strategic priorities.

Mastering this technique ensures that your analytical efforts are directed towards the areas that offer the highest return on investment, fulfilling the core promise of the underlying Pareto Principle: maximizing impact by focusing on the critical few.