Welcome

Aug 3, 2023

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 us 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 Exploratory Univariate and Bivariate Analysis of data and the corresponding Data Visualization.

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

Live Case: We further illustrate how to use R programming in the form of a live project implemented on a real-world dataset. Our dataset concerns the S&P500 stocks. This will demonstrate a practical aspect of using this book. We have many sample codes regarding this, using real-world data.. We will explore financial metrics such as Return on Equity, Return on Assets and Return on Invested Capital of S&P500 shares.

Chapter 1: Getting Started

Chapter 1 provides a comprehensive introduction to the R programming language, focusing on its applications in statistical computations and data analysis. The first part covers the basics of R, including its history, usage, and platforms. It discusses the installation process for R and RStudio, while also mentioning alternatives like Jupyter Notebook and Visual Studio Code. The second part explores practical aspects of using R, demonstrating mathematical and statistical operations with examples. It covers arithmetic, exponentiation, logarithmic operations, and specific mathematical functions like absolute value and square root. The chapter also explains how to compute mean, median, standard deviation, and correlation. It emphasizes the importance and versatility of R in statistical computing, data analysis, and visualization, providing a list of references for further exploration. In summary, the chapter effectively introduces R, its development, installation, and practical applications in statistical computations and data analysis.

Chapter 2: R Packages

Chapter 2 delves into the realm of R packages, elucidating their definition, sources of acquisition, and practical applications. These packages, comprising code, data, and documentation, expand the capabilities of R for users. They can be accessed from repositories like CRAN, Bioconductor, and GitHub, and incorporated into R using the library() function. The advantages of using R packages encompass code reuse, collaborative development, method standardization, handling large datasets, and cross-platform operability. CRAN is highlighted as a pivotal repository, facilitating easy package discovery and installation. The process of installing an R package is explained through the install.packages() function, followed by importing with

library(). The chapter also enumerates notable R packages catering to specific needs, exemplifying ggplot2 for creating visualizations. For addressing challenges, users are directed to package documentation, R's help system, and online resources.