Hands-On Exploratory Data Analysis with Python

Perform EDA techniques to understand, summarize, and investigate your data

Packt

www.packt.com

Suresh Kumar Mukhiya and Usman Ahmed

Hands-On Exploratory Data Analysis with Python

Perform EDA techniques to understand, summarize, and investigate your data

Suresh Kumar Mukhiya
Usman Ahmed

Packt

BIRMINGHAM - MUMBAI

Hands-On Exploratory Data Analysis with Python

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author(s), nor Packt Publishing or its dealers and distributors, will be held liable for any damages caused or alleged to have been caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

Commissioning Editor: Pravin Dhandre
Acquisition Editor: Ali Abidi
Content Development Editor: Nathanya Dias
Senior Editor: Ayaan Hoda
Technical Editor: Manikandan Kurup
Copy Editor: Safis Editing
Project Coordinator: Aishwarya Mohan
Proofreader: Safis Editing
Indexer: Rekha Nair
Production Designer: Deepika Naik

First published: March 2020

Production reference: 1270320

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-78953-725-3

`www.packt.com`

Contributors

About the authors

Suresh Kumar Mukhiya is a Ph.D. candidate currently affiliated with the Western Norway University of Applied Sciences (HVL). He is a big data enthusiast, specializing in information systems, model-driven software engineering, big data analysis, artificial intelligence, and frontend development. He has completed his Master's degree in information systems at the Norwegian University of Science and Technology (NTNU, Norway), along with a thesis in processing mining. He also holds a Bachelor's degree in computer science and information technology (BSc.CSIT) from Tribhuvan University, Nepal, where he was decorated with the Vice-Chancellor's Award for obtaining the highest score. He is a passionate photographer and a resilient traveler.

> *Special thanks go to the people who have helped in the creation of this book. We want to acknowledge the following contributors whose constructive feedback and ideas made this book possible: Asha Gaire (asha.gaire95@gmail.com), Bachelor in Computer Science and Information Technology, Nepal. She proofread the final draft and contributed to the major sections of the book especially Data Transformation, Grouping Dataset, and Correlation chapters. Anju Mukhiya (anjumukhiya@gmail.com) for reading an early draft and making many corrections and suggestions. Lilash Sah, (lilashsah2012@gmail.com) Master in Information Technology, King's Own Institute -Sydney, for reading and validating the codes used in this book.*

Usman Ahmed is a data scientist and Ph.D. candidate at the Western Norway University of Applied Sciences (HVL). He has rich experience in building and scaling high-performance systems based on data mining, natural language processing, and machine learning. Usman's research interests are sequential data mining, heterogeneous computing, natural language processing, recommendation systems, and machine learning. He has completed the Master of Science degree in computer science at Capital University of Science and Technology, Islamabad, Pakistan. Usman Ahmed was awarded a gold medal for his bachelor of computer science degree from Heavy Industries Taxila Education City.

About the reviewer

Jamshaid Sohail is passionate about data science, machine learning, computer vision, natural language processing, and big data, and has completed over 65 online courses in related fields. He has worked in a Silicon Valley-based start-up named Funnelbeam as a data scientist. He worked with the founders of Funnelbeam, who came from Stanford University, and he generated a lot of revenue by completing several projects and products. Currently, he is working as a data scientist at Fiverivers Technologies. He authored the course *Data Wrangling with Python 3.X* for Packt and has reviewed a number of books and courses.

Packt is searching for authors like you

Table of Contents

Section 2: Section 2: Descriptive Statistics

Section 3: Section 3: Model Development and Evaluation

Preface

Data is a collection of discrete objects, events, and facts in the form of numbers, text, pictures, videos, objects, audio, and other entities. Processing data provides a great deal of information. But the million-dollar question is—*how* do we get *meaningful* information from data? The answer to this question is **Exploratory Data Analysis** (**EDA**), which is the process of investigating datasets, elucidating subjects, and visualizing outcomes. EDA is an approach to data analysis that applies a variety of techniques to maximize specific insights into a dataset, reveal an underlying structure, extract significant variables, detect outliers and anomalies, test assumptions, develop models, and determine best parameters for future estimations. This book, *Hands-On Exploratory Data Analysis with Python*, aims to provide practical knowledge about the main pillars of EDA, including data cleansing, data preparation, data exploration, and data visualization. Why visualization? Well, several research studies have shown that portraying data in graphical form makes complex statistical data analyses and business intelligence more marketable.

You will get the opportunity to explore open source datasets including healthcare datasets, demographics datasets, a Titanic dataset, a wine quality dataset, automobile datasets, a Boston housing pricing dataset, and many others. Using these real-life datasets, you will get hands-on practice in understanding data, summarize data's characteristics, and visualizing data for business intelligence purposes. This book expects you to use pandas, a powerful library for working with data, and other core Python libraries including NumPy, scikit-learn, SciPy, StatsModels for regression, and Matplotlib for visualization.

Who this book is for

This book is for anyone who intends to analyze data, including students, teachers, managers, engineers, statisticians, data analysts, and data scientists. The practical concepts presented in this hands-on book are applicable to applications in various disciplines, including linguistics, sociology, astronomy, marketing, business, management, quality control, education, economics, medicine, psychology, engineering, biology, physics, computer science, geosciences, chemistry, and any other fields where data analysis and synthesis is required in order to improve knowledge and help in decision-making processes. Fundamental understanding of Python programming and some statistical concepts is all you need to get started with this book.

What this book covers

`Chapter 1`, *Exploratory Data Analysis Fundamentals*, will help us learn and revise the fundamental aspects of EDA. We will dig into the importance of EDA and the main data analysis tasks, and try to make sense out of data. In addition to that, we will use Python to explore different types of data, including numerical data, time-series data, geospatial data, categorical data, and others.

`Chapter 2`, *Visual Aids for EDA*, will help us gain proficiency with different tools for visualizing the information that we get from investigation and make analysis much clearer. We will figure out how to use data visualization tools such as box plots, histograms, multi-variate charts, and more. Notwithstanding that, we will get our hands dirty in plotting an enlightening visual graph using real databases. Finally, we will investigate the intuitive forms of these plots.

`Chapter 3`, *EDA with Personal Email*, will help us figure out how to import a dataset from your personal Gmail account and work on analyzing the extracted dataset. We will perform basic EDA techniques, including data loading, data cleansing, data preparation, data visualization, and data analysis, on the extracted dataset.

`Chapter 4`, *Data Transformation*, is where you will take your first steps in data wrangling. We will see how to merge database-style DataFrames, merge on the index, concatenate along an axis, combine data with overlaps, reshape with hierarchical indexing, and pivot from long to wide format. We will look at what needs to be done with a dataset before analysis takes place, such as removing duplicates, replacing values, renaming axis indexes, discretization and binning, and detecting and filtering outliers. We will work on transforming data using a function or mapping, permutation, and random sampling and computing indicators/dummy variables.

`Chapter 5`, *Descriptive Statistics*, will teach you about essential statistical measures for gaining insights about data that are not noticeable at the surface level. We will become familiar with the equations for computing the variance and standard deviation of datasets as well as figuring out percentiles and quartiles. Furthermore, we will envision those factual measures with visualization. We will use tools such as box plots to gain knowledge from statistics.

`Chapter 6`, *Grouping Datasets*, will cover the rudiments of grouping and how it can change our datasets in order to help us to analyze them better. We will look at different group-by mechanics that will amass our dataset into various classes in which we can perform aggregate activities. We will also figure out how to dissect categorical data with visualizations, utilizing pivot tables and cross-tabulations.

`Chapter 7`, *Correlation*, will help us to understand the correlation between different factors and to identify to what degree different factors are relevant. We will learn about the different kinds of examinations that we can carry out to discover the relationships between data, including univariate analysis, bivariate analysis, and multivariate analysis on the Titanic dataset, as well as looking at Simpson's paradox. We will observe how correlation does not always equal causation.

`Chapter 8`, *Time Series Analysis*, will help us to understand time-series data and how to perform EDA on it. We will use the open power system data for time series analysis.

`Chapter 9`, *Hypothesis Testing and Regression*, will help us learn about hypothesis testing and linear, non-linear, and multiple linear regression. We will build a basis for model development and evaluation. We will be using polynomial regression and pipelines for model evaluation.

`Chapter 10`, *Model Development and Evaluation*, will help us learn about a unified machine learning approach, discuss different types of machine learning algorithms and evaluation techniques. Moreover, in this chapter, we are going to perform the unsupervised learning task of clustering with text data. Furthermore, we will discuss model selection and model deployment techniques.

`Chapter 11`, *EDA on Wine Quality Data*, will teach us how to use all the techniques learned throughout the book to perform advanced EDA on a wine quality dataset. We will import the dataset, research the variables, slice the data based on different points of interest, and perform data analysis.

To get the most out of this book

All the EDA activities in this book are based on Python 3.x. So, the first and foremost requirement to run any code from this book is for you to have Python 3.x installed on your computer irrespective of the operating system. Python can be installed on your system by following the documentation on its official website: `https://www.python.org/downloads/`.

Here is the software that needs to be installed in order to execute the code:

Software/hardware covered in the book	OS requirements
Python 3.x	Windows, macOS, Linux, or any other OS
Python notebooks	There are several options: Local: Jupyter: `https://jupyter.org/` Local: `https://www.anaconda.com/distribution/` Online: `https://colab.research.google.com/`
Python libraries	NumPy, pandas, scikit-learn, Matplotlib, Seaborn, StatsModel

We primarily used Python notebooks to execute our code. One of the reasons for that is, with them, it is relatively easy to break code into a clear structure and see the output on the fly. It is always safer to install a notebook locally. The official website holds great information on how they can be installed. However, if you do not want the hassle and simply want to start learning immediately, then Google Colab provides a great platform where you can code and execute code using both Python 2.x and Python 3.x with support for **Graphics Processing Units** (**GPUs**) and **Tensor Processing Units** (**TPUs**).

If you are using the digital version of this book, we advise you to type the code yourself or access the code via the GitHub repository (link available in the next section). Doing so will help you avoid any potential errors related to the copying and pasting of code.

Download the example code files

You can download the example code files for this book from your account at `www.packt.com`. If you purchased this book elsewhere, you can visit `www.packtpub.com/support` and register to have the files emailed directly to you.

You can download the code files by following these steps:

1. Log in or register at `www.packt.com`.
2. Select the **Support** tab.
3. Click on **Code Downloads**.
4. Enter the name of the book in the **Search** box and follow the onscreen instructions.

Once the file is downloaded, please make sure that you unzip or extract the folder using the latest version of:

- WinRAR/7-Zip for Windows
- Zipeg/iZip/UnRarX for Mac
- 7-Zip/PeaZip for Linux

The code bundle for the book is also hosted on GitHub at `https://github.com/PacktPublishing/hands-on-exploratory-data-analysis-with-python`. In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at `https://github.com/PacktPublishing/`. Check them out!

Download the color images

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: `https://static.packt-cdn.com/downloads/9781789537253_ColorImages.pdf`.

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in the text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "we visualized a time series dataset using the `matplotlib` and `seaborn` libraries."

A block of code is set as follows:

```
import os
import numpy as np
%matplotlib inline from matplotlib
import pyplot as plt
import seaborn as sns
```

Any command-line input or output is written as follows:

```
> pip install virtualenv
> virtualenv Local_Version_Directory -p Python_System_Directory
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Time series data may contain a notable amount of **outliers**."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at `customercare@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packtpub.com/support/errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packt.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit `packt.com`.

1

Section 1: The Fundamentals of EDA

The main objective of this section is to cover the fundamentals of **Exploratory Data Analysis** (**EDA**) and understand different stages of the EDA process. We will also look at the key concepts of profiling, quality assessment, the main aspects of EDA, and the challenges and opportunities in EDA. In addition to this, we will be discovering different useful visualization techniques. Finally, we will be discussing essential data transformation techniques, including database-style dataframe merges, transformation techniques, and benefits of data transformation.

This section contains the following chapters:

- `Chapter 1`, *Exploratory Data Analysis Fundamentals*
- `Chapter 2`, *Visual Aids for EDA*
- `Chapter 3`, *EDA with Personal Email*
- `Chapter 4`, *Data Transformation*

1
Exploratory Data Analysis Fundamentals

The main objective of this introductory chapter is to revise the fundamentals of **Exploratory Data Analysis** (**EDA**), what it is, the key concepts of profiling and quality assessment, the main dimensions of EDA, and the main challenges and opportunities in EDA.

Data encompasses a collection of discrete objects, numbers, words, events, facts, measurements, observations, or even descriptions of things. Such data is collected and stored by every event or process occurring in several disciplines, including biology, economics, engineering, marketing, and others. Processing such data elicits useful *information* and processing such information generates useful knowledge. But an important question is: how can we generate meaningful and useful information from such data? An answer to this question is EDA. EDA is a process of examining the available dataset to discover patterns, spot anomalies, test hypotheses, and check assumptions using statistical measures. In this chapter, we are going to discuss the steps involved in performing top-notch exploratory data analysis and get our hands dirty using some open source databases.

As mentioned here and in several studies, the primary aim of EDA is to examine what data can tell us before actually going through formal modeling or hypothesis formulation. John Tuckey promoted EDA to statisticians to examine and discover the data and create newer hypotheses that could be used for the development of a newer approach in data collection and experimentations.

In this chapter, we are going to learn and revise the following topics:

- Understanding data science
- The significance of EDA
- Making sense of data
- Comparing EDA with classical and Bayesian analysis
- Software tools available for EDA
- Getting started with EDA

Understanding data science

Let's get this out of the way by pointing out that, if you have not heard about data science, then you should not be reading this book. Everyone right now is talking about data science in one way or another. Data science is at the peak of its hype and the skills for data scientists are changing. Now, data scientists are not only required to build a performant model, but it is essential for them to explain the results obtained and use the result for business intelligence. During my talks, seminars, and presentations, I find several people trying to ask me: *what type of skillset do I need to learn in order to become a top-notch data scientist? Do I need to get a Ph.D. in data science?* Well, one thing I could tell you straight away is you do not need a Ph.D. to be an expert in data science. But one thing that people generally agree on is that data science involves cross-disciplinary knowledge from computer science, data, statistics, and mathematics. There are several phases of data analysis, including data requirements, data collection, data processing, data cleaning, exploratory data analysis, modeling and algorithms, and data product and communication. These phases are similar to the **CRoss-Industry Standard Process for data mining** (**CRISP**) framework in data mining.

The main takeaway here is the stages of EDA, as it is an important aspect of data analysis and data mining. Let's understand in brief what these stages are:

- **Data requirements:** There can be various sources of data for an organization. It is important to comprehend what type of data is required for the organization to be collected, curated, and stored. For example, an application tracking the sleeping pattern of patients suffering from dementia requires several types of sensors' data storage, such as sleep data, heart rate from the patient, electro-dermal activities, and user activities pattern. All of these data points are required to correctly diagnose the mental state of the person. Hence, these are mandatory requirements for the application. In addition to this, it is required to categorize the data, numerical or categorical, and the format of storage and dissemination.
- **Data collection:** Data collected from several sources must be stored in the correct format and transferred to the right information technology personnel within a company. As mentioned previously, data can be collected from several objects on several events using different types of sensors and storage tools.
- **Data processing:** Preprocessing involves the process of pre-curating the dataset before actual analysis. Common tasks involve correctly exporting the dataset, placing them under the right tables, structuring them, and exporting them in the correct format.

- **Data cleaning:** Preprocessed data is still not ready for detailed analysis. It must be correctly transformed for an incompleteness check, duplicates check, error check, and missing value check. These tasks are performed in the data cleaning stage, which involves responsibilities such as matching the correct record, finding inaccuracies in the dataset, understanding the overall data quality, removing duplicate items, and filling in the missing values. However, how could we identify these anomalies on any dataset? Finding such data issues requires us to perform some analytical techniques. We will be learning several such analytical techniques in `Chapter 4`, *Data Transformation*. To understand briefly, data cleaning is dependent on the types of data under study. Hence, it is most essential for data scientists or EDA experts to comprehend different types of datasets. An example of data cleaning would be using outlier detection methods for quantitative data cleaning.
- **EDA:** Exploratory data analysis, as mentioned before, is the stage where we actually start to understand the message contained in the data. It should be noted that several types of data transformation techniques might be required during the process of exploration. We will cover descriptive statistics in-depth in *Section 2*, `Chapter 5`, *Descriptive Statistics*, to understand the mathematical foundation behind descriptive statistics. This entire book is dedicated to tasks involved in exploratory data analysis.
- **Modeling and algorithm**: From a data science perspective, generalized models or mathematical formulas can represent or exhibit relationships among different variables, such as correlation or causation. These models or equations involve one or more variables that depend on other variables to cause an event. For example, when buying, say, pens, the total price of *pens(Total) = price for one pen(UnitPrice) * the number of pens bought (Quantity)*. Hence, our model would be *Total = UnitPrice * Quantity*. Here, the total price is dependent on the unit price. Hence, the total price is referred to as the dependent variable and the unit price is referred to as an independent variable. In general, a model always describes the relationship between independent and dependent variables. Inferential statistics deals with quantifying relationships between particular variables.
  The Judd model for describing the relationship between data, model, and error still holds true: *Data = Model + Error.* We will discuss in detail model development in *Section 3*, `Chapter 10`, *Model Evaluation*. An example of inferential statistics would be regression analysis. We will discuss regression analysis in `Chapter 9`, *Regression*.

- **Data Product:** Any computer software that uses data as inputs, produces outputs, and provides feedback based on the output to control the environment is referred to as a data product. A data product is generally based on a model developed during data analysis, for example, a recommendation model that inputs user purchase history and recommends a related item that the user is highly likely to buy.
- **Communication:** This stage deals with disseminating the results to end stakeholders to use the result for *business intelligence*. One of the most notable steps in this stage is data visualization. Visualization deals with information relay techniques such as tables, charts, summary diagrams, and bar charts to show the analyzed result. We will outline several visualization techniques in `Chapter 2`, *Visual Aids for EDA*, with different types of data.

The significance of EDA

Different fields of science, economics, engineering, and marketing accumulate and store data primarily in electronic databases. Appropriate and well-established decisions should be made using the data collected. It is practically impossible to make sense of datasets containing more than a handful of data points without the help of computer programs. To be certain of the insights that the collected data provides and to make further decisions, data mining is performed where we go through distinctive analysis processes. Exploratory data analysis is key, and usually the first exercise in data mining. It allows us to visualize data to understand it as well as to create hypotheses for further analysis. The exploratory analysis centers around creating a synopsis of data or insights for the next steps in a data mining project.

EDA actually reveals ground truth about the content without making any underlying assumptions. This is the fact that data scientists use this process to actually understand what type of modeling and hypotheses can be created. Key components of exploratory data analysis include summarizing data, statistical analysis, and visualization of data. Python provides expert tools for exploratory analysis, with `pandas` for summarizing; `scipy`, along with others, for statistical analysis; and `matplotlib` and `plotly` for visualizations.

That makes sense, right? Of course it does. That is one of the reasons why you are going through this book. After understanding the significance of EDA, let's discover what are the most generic steps involved in EDA in the next section.

Steps in EDA

Having understood what EDA is, and its significance, let's understand the various steps involved in data analysis. Basically, it involves four different steps. Let's go through each of them to get a brief understanding of each step:

- **Problem definition:** Before trying to extract useful insight from the data, it is essential to define the business problem to be solved. The problem definition works as the driving force for a data analysis plan execution. The main tasks involved in problem definition are defining the main objective of the analysis, defining the main deliverables, outlining the main roles and responsibilities, obtaining the current status of the data, defining the timetable, and performing cost/benefit analysis. Based on such a problem definition, an execution plan can be created.
- **Data preparation**: This step involves methods for preparing the dataset before actual analysis. In this step, we define the sources of data, define data schemas and tables, understand the main characteristics of the data, clean the dataset, delete non-relevant datasets, transform the data, and divide the data into required chunks for analysis.
- **Data analysis:** This is one of the most crucial steps that deals with descriptive statistics and analysis of the data. The main tasks involve summarizing the data, finding the hidden correlation and relationships among the data, developing predictive models, evaluating the models, and calculating the accuracies. Some of the techniques used for data summarization are summary tables, graphs, descriptive statistics, inferential statistics, correlation statistics, searching, grouping, and mathematical models.
- **Development and representation of the results:** This step involves presenting the dataset to the target audience in the form of graphs, summary tables, maps, and diagrams. This is also an essential step as the result analyzed from the dataset should be interpretable by the business stakeholders, which is one of the major goals of EDA. Most of the graphical analysis techniques include scattering plots, character plots, histograms, box plots, residual plots, mean plots, and others. We will explore several types of graphical representation in `Chapter 2`, *Visual Aids for EDA*.

Making sense of data

It is crucial to identify the type of data under analysis. In this section, we are going to learn about different types of data that you can encounter during analysis. Different disciplines store different kinds of data for different purposes. For example, medical researchers store patients' data, universities store students' and teachers' data, and real estate industries storehouse and building datasets. A dataset contains many observations about a particular object. For instance, a dataset about patients in a hospital can contain many observations. A patient can be described by a *patient identifier (ID), name, address, weight, date of birth, address, email,* and *gender*. Each of these features that describes a patient is a variable. Each observation can have a specific value for each of these variables. For example, a patient can have the following:

```
PATIENT_ID = 1001
Name = Yoshmi Mukhiya
Address = Mannsverk 61, 5094, Bergen, Norway
Date of birth = 10th July 2018
Email = yoshmimukhiya@gmail.com
Weight = 10
Gender = Female
```

These datasets are stored in hospitals and are presented for analysis. Most of this data is stored in some sort of database management system in tables/schema. An example of a table for storing patient information is shown here:

PATIENT_ID	NAME	ADDRESS	DOB	EMAIL	Gender	WEIGHT
001	Suresh Kumar Mukhiya	Mannsverk, 61	30.12.1989	skmu@hvl.no	Male	68
002	Yoshmi Mukhiya	Mannsverk 61, 5094, Bergen	10.07.2018	yoshmimukhiya@gmail.com	Female	1
003	Anju Mukhiya	Mannsverk 61, 5094, Bergen	10.12.1997	anjumukhiya@gmail.com	Female	24
004	Asha Gaire	Butwal, Nepal	30.11.1990	aasha.gaire@gmail.com	Female	23
005	Ola Nordmann	Danmark, Sweden	12.12.1789	ola@gmail.com	Male	75

To summarize the preceding table, there are four observations (001, 002, 003, 004, 005). Each observation describes variables (`PatientID`, `name`, `address`, `dob`, `email`, `gender`, and `weight`). Most of the dataset broadly falls into two groups—numerical data and categorical data.

Numerical data

This data has a sense of measurement involved in it; for example, a person's age, height, weight, blood pressure, heart rate, temperature, number of teeth, number of bones, and the number of family members. This data is often referred to as **quantitative data** in statistics. The numerical dataset can be either discrete or continuous types.

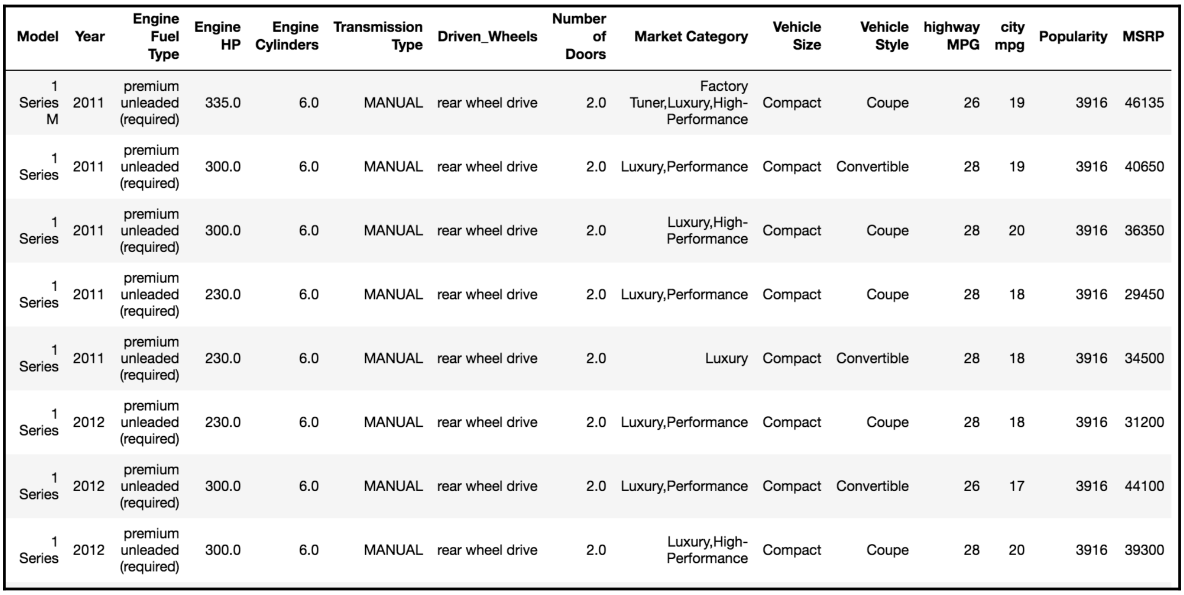
Discrete data

This is data that is countable and its values can be listed out. For example, if we flip a coin, the number of heads in 200 coin flips can take values from 0 to 200 (finite) cases. A variable that represents a discrete dataset is referred to as a discrete variable. The discrete variable takes a fixed number of distinct values. For example, the `Country` variable can have values such as Nepal, India, Norway, and Japan. It is fixed. The `Rank` variable of a student in a classroom can take values from 1, 2, 3, 4, 5, and so on.

Continuous data

A variable that can have an infinite number of numerical values within a specific range is classified as continuous data. A variable describing continuous data is a continuous variable. For example, what is the temperature of your city today? Can we be finite? Similarly, the `weight` variable in the previous section is a continuous variable. We are going to use a car dataset in `Chapter 5`, *Descriptive Statistics*, to perform EDA.

A section of the table is shown in the following table:



Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popularity	MSRP
1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	46135
1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	40650
1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	36350
1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	29450
1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	34500
1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	31200
1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	26	17	3916	44100
1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	39300

Check the preceding table and determine which of the variables are discrete and which of the variables are continuous. Can you justify your claim? Continuous data can follow an interval measure of scale or ratio measure of scale. We will go into more detail in the *Measurement scales* section in this chapter.

Categorical data

This type of data represents the characteristics of an object; for example, gender, marital status, type of address, or categories of the movies. This data is often referred to as **qualitative datasets** in statistics. To understand clearly, here are some of the most common types of categorical data you can find in data:

- Gender (Male, Female, Other, or Unknown)
- Marital Status (Annulled, Divorced, Interlocutory, Legally Separated, Married, Polygamous, Never Married, Domestic Partner, Unmarried, Widowed, or Unknown)
- Movie genres (Action, Adventure, Comedy, Crime, Drama, Fantasy, Historical, Horror, Mystery, Philosophical, Political, Romance, Saga, Satire, Science Fiction, Social, Thriller, Urban, or Western)

- Blood type (A, B, AB, or O)
- Types of drugs (Stimulants, Depressants, Hallucinogens, Dissociatives, Opioids, Inhalants, or Cannabis)

A variable describing categorical data is referred to as a **categorical variable**. These types of variables can have one of a limited number of values. It is easier for computer science students to understand categorical values as enumerated types or enumerations of variables. There are different types of categorical variables:

- A binary categorical variable can take exactly two values and is also referred to as a **dichotomous variable**. For example, when you create an experiment, the result is either success or failure. Hence, results can be understood as a **binary categorical variable**.
- **Polytomous variables** are categorical variables that can take more than two possible values. For example, marital status can have several values, such as annulled, divorced, interlocutory, legally separated, married, polygamous, never married, domestic partners, unmarried, widowed, domestic partner, and unknown. Since marital status can take more than two possible values, it is a **polytomous variable.**

Most of the categorical dataset follows either nominal or ordinal measurement scales. Let's understand what is a nominal or ordinal scale in the next section.

Measurement scales

There are four different types of measurement scales described in statistics: nominal, ordinal, interval, and ratio. These scales are used more in academic industries. Let's understand each of them with some examples.

Nominal

These are practiced for labeling variables without any quantitative value. The scales are generally referred to as **labels**. And these scales are mutually exclusive and do not carry any numerical importance. Let's see some examples:

- What is your gender?
  - Male
  - Female
  - Third gender/Non-binary

    - I prefer not to answer
    - Other
- Other examples include the following:
    - The languages that are spoken in a particular country
    - Biological species
    - Parts of speech in grammar (noun, pronoun, adjective, and so on)
    - Taxonomic ranks in biology (Archea, Bacteria, and Eukarya)

Nominal scales are considered qualitative scales and the measurements that are taken using qualitative scales are considered **qualitative data**. However, the advancement in qualitative research has created confusion to be definitely considered as qualitative. If, for example, someone uses numbers as labels in the nominal measurement sense, they have no concrete numerical value or meaning. No form of arithmetic calculation can be made on nominal measures.

You might be thinking *why should you care about whether data is nominal or ordinal? Should we not just start loading the data and begin our analysis?* Well, we could. But think about this: you have a dataset, and you want to analyze it. How will you decide whether you can make a pie chart, bar chart, or histogram? Are you getting my point?

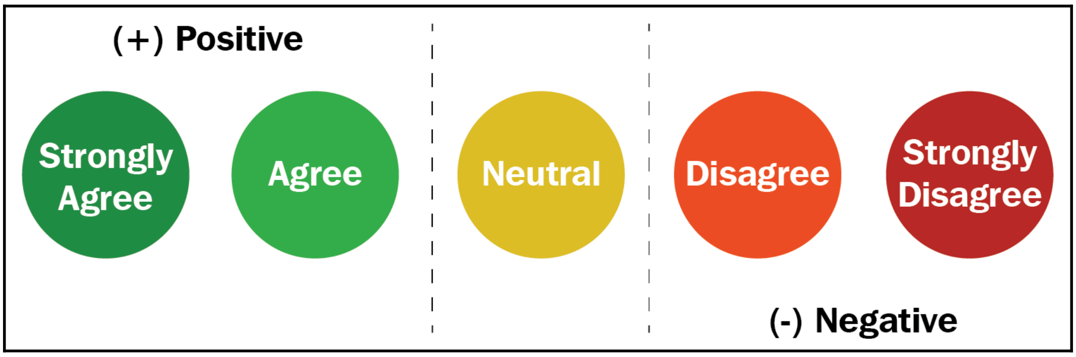
Well, for example, in the case of a nominal dataset, you can certainly know the following:

- **Frequency** is the rate at which a label occurs over a period of time within the dataset.
- **Proportion** can be calculated by dividing the frequency by the total number of events.
- Then, you could compute the **percentage** of each proportion.
- And to **visualize** the nominal dataset, you can use either a pie chart or a bar chart.

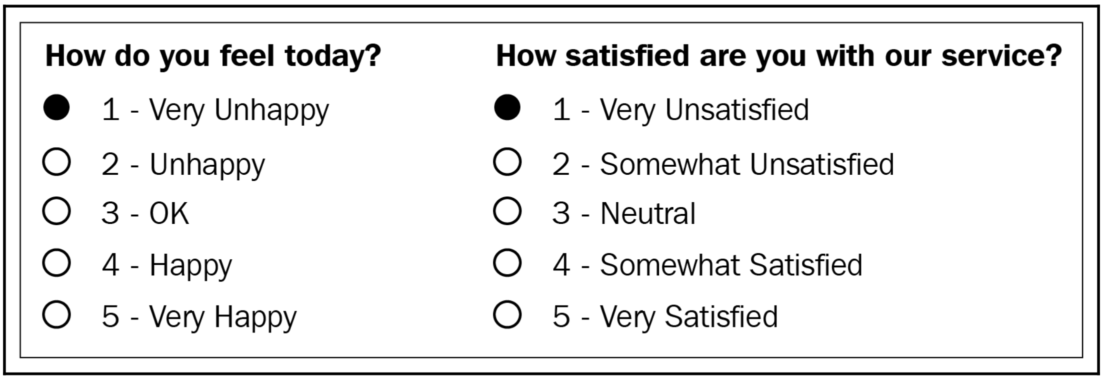
If you know your data follows nominal scales, you can use a pie chart or bar chart. That's one less thing to worry about, right? My point is, understanding the type of data is relevant in understanding what type of computation you can perform, what type of model you should fit on the dataset, and what type of visualization you can generate.

Ordinal

The main difference in the ordinal and nominal scale is the order. In ordinal scales, the order of the values is a significant factor. An easy tip to remember the ordinal scale is that it sounds like an *order*. Have you heard about the **Likert scale**, which uses a variation of an ordinal scale? Let's check an example of ordinal scale using the Likert scale: *WordPress is making content managers' lives easier. How do you feel about this statement?* The following diagram shows the Likert scale:



As depicted in the preceding diagram, the answer to the question of *WordPress is making content managers' lives easier* is scaled down to five different ordinal values, **Strongly Agree**, **Agree**, **Neutral**, **Disagree**, and **Strongly Disagree**. Scales like these are referred to as the Likert scale. Similarly, the following diagram shows more examples of the Likert scale:



How do you feel today?

● 1 - Very Unhappy
○ 2 - Unhappy
○ 3 - OK
○ 4 - Happy
○ 5 - Very Happy

How satisfied are you with our service?

● 1 - Very Unsatisfied
○ 2 - Somewhat Unsatisfied
○ 3 - Neutral
○ 4 - Somewhat Satisfied
○ 5 - Very Satisfied

To make it easier, consider ordinal scales as an order of ranking (1st, 2nd, 3rd, 4th, and so on). The **median** item is allowed as the measure of central tendency; however, the **average** is not permitted.

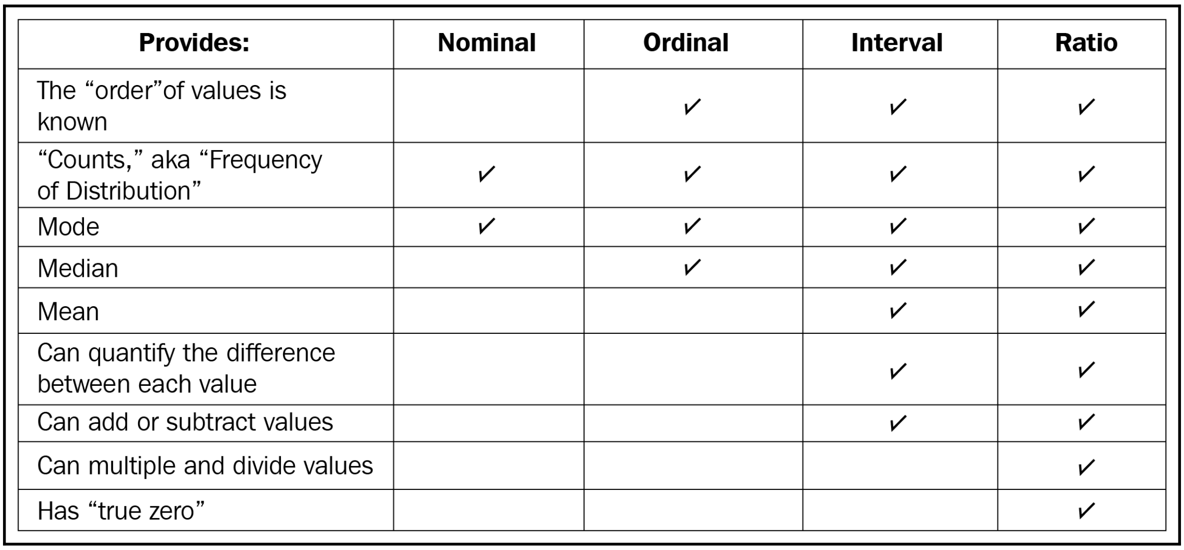
Interval

In interval scales, both the order and exact differences between the values are significant. Interval scales are widely used in statistics, for example, in the *measure of central tendencies—mean, median, mode, and standard deviations.* Examples include location in Cartesian coordinates and direction measured in degrees from magnetic north. The mean, median, and mode are allowed on interval data.

Ratio

Ratio scales contain order, exact values, and absolute zero, which makes it possible to be used in descriptive and inferential statistics. These scales provide numerous possibilities for statistical analysis. Mathematical operations, the measure of central tendencies, and the **measure of dispersion** and **coefficient of variatio**n can also be computed from such scales.

Examples include a measure of energy, mass, length, duration, electrical energy, plan angle, and volume. The following table gives a summary of the data types and scale measures:



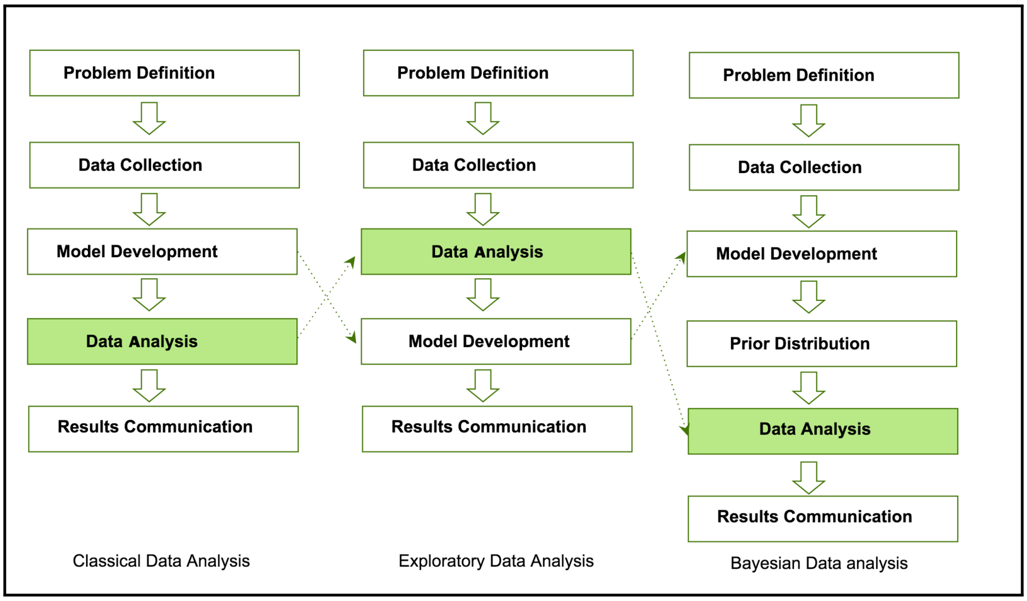
Provides:	Nominal	Ordinal	Interval	Ratio
The “order”of values is known		✓	✓	✓
“Counts,” aka “Frequency of Distribution”	✓	✓	✓	✓
Mode	✓	✓	✓	✓
Median		✓	✓	✓
Mean			✓	✓
Can quantify the difference between each value			✓	✓
Can add or subtract values			✓	✓
Can multiple and divide values				✓
Has “true zero”				✓

In the next section, we will compare EDA with classical and Bayesian analysis.

Comparing EDA with classical and Bayesian analysis

There are several approaches to data analysis. The most popular ones that are relevant to this book are the following:

- **Classical data analysis:** For the classical data analysis approach, the problem definition and data collection step are followed by model development, which is followed by analysis and result communication.
- **Exploratory data analysis approach**: For the EDA approach, it follows the same approach as classical data analysis except the model imposition and the data analysis steps are swapped. The main focus is on the data, its structure, outliers, models, and visualizations. Generally, in EDA, we do not impose any deterministic or probabilistic models on the data.
- **Bayesian data analysis approach:** The Bayesian approach incorporates prior probability distribution knowledge into the analysis steps as shown in the following diagram. Well, simply put, prior probability distribution of any quantity expresses the belief about that particular quantity before considering some evidence. Are you still lost with the term prior probability distribution? Andrew Gelman has a very descriptive paper about *prior probability distribution*. The following diagram shows three different approaches for data analysis illustrating the difference in their execution steps:



Data analysts and data scientists freely mix steps mentioned in the preceding approaches to get meaningful insights from the data. In addition to that, it is essentially difficult to judge or estimate which model is best for data analysis. All of them have their paradigms and are suitable for different types of data analysis.

Software tools available for EDA

There are several software tools that are available to facilitate EDA. Here, we are going to outline some of the open source tools:

- **Python**: This is an open source programming language widely used in data analysis, data mining, and data science (`https://www.python.org/`). For this book, we will be using Python.
- **R programming language**: R is an open source programming language that is widely utilized in statistical computation and graphical data analysis (`https://www.r-project.org`).
- **Weka**: This is an open source data mining package that involves several EDA tools and algorithms (`https://www.cs.waikato.ac.nz/ml/weka/`).
- **KNIME**: This is an open source tool for data analysis and is based on Eclipse (`https://www.knime.com/`).

Getting started with EDA

As mentioned earlier, we are going to use Python as the main tool for data analysis. Yay! Well, if you ask me why, Python has been consistently ranked among the top 10 programming languages and is widely adopted for data analysis and data mining by data science experts. In this book, we assume you have a working knowledge of Python. If you are not familiar with Python, it's probably too early to get started with data analysis. I assume you are familiar with the following Python tools and packages:

Python programming	Fundamental concepts of variables, string, and data types Conditionals and functions Sequences, collections, and iterations Working with files Object-oriented programming

NumPy	Create arrays with NumPy, copy arrays, and divide arrays Perform different operations on NumPy arrays Understand array selections, advanced indexing, and expanding Working with multi-dimensional arrays Linear algebraic functions and built-in NumPy functions
pandas	Understand and create `DataFrame` objects Subsetting data and indexing data Arithmetic functions, and mapping with pandas Managing index Building style for visual analysis
Matplotlib	Loading linear datasets Adjusting axes, grids, labels, titles, and legends Saving plots
SciPy	Importing the package Using statistical packages from SciPy Performing descriptive statistics Inference and data analysis

Before diving into details about analysis, we need to make sure we are on the same page. Let's go through the checklist and verify that you meet all of the prerequisites to get the best out of this book:

<table>
<tr><td>Setting up a virtual environment</td><td><pre>> <b>pip install virtualenv</b>
> <b>virtualenv Local_Version_Directory -p Python_System_Directory</b></pre></td></tr>
<tr><td>Reading/writing to files</td><td><pre>filename = "datamining.txt"
file = open(filename, mode="r", encoding='utf-8')
for line in file:
 lines = file.readlines()
print(lines)
file.close()</pre></td></tr>
<tr><td>Error handling</td><td><pre>try:
  Value = int(input("Type a number between 47 and 100:"))
except ValueError:
   print("You must type a number between 47 and 100!")
else:
   if (Value > 47) and (Value <= 100):
       print("You typed: ", Value)
   else:
       print("The value you typed is incorrect!")</pre></td></tr>
<tr><td>Object-oriented concept</td><td><pre>class Disease:
  def __init__(self, disease = 'Depression'):
    self.type = disease
  def getName(self):
    print("Mental Health Diseases: {0}".format(self.type))

d1 = Disease('Social Anxiety Disorder')
d1.getName()</pre></td></tr>
</table>

Next, let's look at the basic operations of EDA using the NumPy library.

NumPy

In this section, we are going to revise the basic operations of EDA using the NumPy library. If you are familiar with these operations, feel free to jump to the next section. It might feel obvious when going through the code, but it is essential to make sure you understand these concepts before digging into EDA operations. When I started learning data science approaches, I followed a lot of blogs where they just reshaped an array or matrix. When I ran their code, it worked fine, but I never understood how I was able to add two matrices of different dimensions. In this section, I have tried to explicitly point out some of the basic numpy operations:

- For importing numpy, we will use the following code:

```
import numpy as np
```

- For creating different types of numpy arrays, we will use the following code:

```
# importing numpy
import numpy as np

# Defining 1D array
my1DArray = np.array([1, 8, 27, 64])
print(my1DArray)

# Defining and printing 2D array
my2DArray = np.array([[1, 2, 3, 4], [2, 4, 9, 16], [4, 8, 18, 32]])
print(my2DArray)

#Defining and printing 3D array
my3Darray = np.array([[[ 1, 2 , 3 , 4],[ 5 , 6 , 7 ,8]], [[ 1, 2,
3, 4],[ 9, 10, 11, 12]]])
print(my3Darray)
```

- For displaying basic information, such as the data type, shape, size, and strides of a NumPy array, we will use the following code:

```
# Print out memory address
print(my2DArray.data)

# Print the shape of array
print(my2DArray.shape)

# Print out the data type of the array
print(my2DArray.dtype)

# Print the stride of the array.
print(my2DArray.strides)
```

- For creating an array using built-in NumPy functions, we will use the following code:

```
# Array of ones
ones = np.ones((3,4))
print(ones)

# Array of zeros
zeros = np.zeros((2,3,4),dtype=np.int16)
print(zeros)

# Array with random values
np.random.random((2,2))

# Empty array
emptyArray = np.empty((3,2))
print(emptyArray)

# Full array
fullArray = np.full((2,2),7)
print(fullArray)

# Array of evenly-spaced values
evenSpacedArray = np.arange(10,25,5)
print(evenSpacedArray)

# Array of evenly-spaced values
evenSpacedArray2 = np.linspace(0,2,9)
print(evenSpacedArray2)
```

- For NumPy arrays and file operations, we will use the following code:

```
# Save a numpy array into file
x = np.arange(0.0,50.0,1.0)
np.savetxt('data.out', x, delimiter=',')

# Loading numpy array from text
z = np.loadtxt('data.out', unpack=True)
print(z)

# Loading numpy array using genfromtxt method
my_array2 = np.genfromtxt('data.out',
                      skip_header=1,
                      filling_values=-999)
print(my_array2)
```

- For inspecting NumPy arrays, we will use the following code:

```
# Print the number of `my2DArray`'s dimensions
print(my2DArray.ndim)

# Print the number of `my2DArray`'s elements
print(my2DArray.size)

# Print information about `my2DArray`'s memory layout
print(my2DArray.flags)

# Print the length of one array element in bytes
print(my2DArray.itemsize)

# Print the total consumed bytes by `my2DArray`'s elements
print(my2DArray.nbytes)
```

- Broadcasting is a mechanism that permits NumPy to operate with arrays of different shapes when performing arithmetic operations:

```
# Rule 1: Two dimensions are operatable if they are equal
# Create an array of two dimension
A =np.ones((6, 8))

# Shape of A
print(A.shape)

# Create another array
B = np.random.random((6,8))

# Shape of B
print(B.shape)

# Sum of A and B, here the shape of both the matrix is same.
print(A + B)
```

Secondly, two dimensions are also compatible when one of the dimensions of the array is 1. Check the example given here:

```
# Rule 2: Two dimensions are also compatible when one of them is 1
# Initialize `x`
x = np.ones((3,4))
print(x)

# Check shape of `x`
print(x.shape)

# Initialize `y`
y = np.arange(4)
print(y)

# Check shape of `y`
print(y.shape)

# Subtract `x` and `y`
print(x - y)
```

Lastly, there is a third rule that says two arrays can be broadcast together if they are compatible in all of the dimensions. Check the example given here:

```
# Rule 3: Arrays can be broadcast together if they are compatible
in all dimensions
x = np.ones((6,8))
y = np.random.random((10, 1, 8))
print(x + y)
```

The dimensions of *x(6,8)* and *y(10,1,8)* are different. However, it is possible to add them. Why is that? Also, change *y(10,2,8)* or *y(10,1,4)* and it will give `ValueError`. Can you find out why? (**Hint**: check rule 1).

- For seeing NumPy mathematics at work, we will use the following example:

```
# Basic operations (+, -, *, /, %)
x = np.array([[1, 2, 3], [2, 3, 4]])
y = np.array([[1, 4, 9], [2, 3, -2]])

# Add two array
add = np.add(x, y)
print(add)

# Subtract two array
sub = np.subtract(x, y)
print(sub)

# Multiply two array
mul = np.multiply(x, y)
print(mul)

# Divide x, y
div = np.divide(x,y)
print(div)

# Calculated the remainder of x and y
rem = np.remainder(x, y)
print(rem)
```

- Let's now see how we can create a subset and slice an array using an index:

```
x = np.array([10, 20, 30, 40, 50])

# Select items at index 0 and 1
print(x[0:2])

# Select item at row 0 and 1 and column 1 from 2D array
y = np.array([[ 1, 2, 3, 4], [ 9, 10, 11 ,12]])
print(y[0:2, 1])

# Specifying conditions
biggerThan2 = (y >= 2)
print(y[biggerThan2])
```

Next, we will use the `pandas` library to gain insights from data.