Chapter: 04

EXPLORATORY DATA ANALYSIS USING PYTHON

Mohd Hyder Gouri*
Faculty, Glocal School of Science and Technology,
Glocal University, Saharanpur, U.P.
**Correspondence to: hyder@theglocaluniversity.in*

Mohd Nafees
Faculty, Glocal School of Science and Technology,
Glocal University, Saharanpur, U.P.

DOI: https://doi.org/10.52458/9789388996747.nsp2023.eb.ch-04

Ch.Id:-GU/NSP/EB/EFMLDSP/2023/Ch-04

ABSTRACT

Data scientists and analysts can analyze, display, and get important insights from their datasets through exploratory data analysis (EDA), a critical phase in the data analysis process. With its extensive data manipulation and visualization module ecosystem, Python has become a potent tool in this situation. This chapter provides a thorough introduction of Python-based EDA techniques, highlighting the value of EDA in the pipeline for data analysis and presenting different approaches to data visualization, summary statistics, and statistical testing.

Keywords: *Exploratory Data Analysis, EDA, Python, Data Analysis, Data Visualization, Summary Statistics, Data Exploration, Data Insights, Data Science*

4.1 INTRODUCTION

The growing accessibility of huge and complicated datasets has caused a paradigm change in the field of data analysis. Understanding your data is essential in the big data age we live in today. Data analysis begins with exploratory data analysis (EDA), which acts as a compass for navigating the data world. It entails methodically going through, summarizing, and visualizing data in order to draw out important conclusions and spot trends. Python is now the most popular choice for doing EDA because of its vast libraries and interactive features. Python is a flexible and widely used programming language.

This chapter intends to offer a thorough overview of Python-based EDA. For effective data exploration, it will include a range of methods, resources, and best practices. In addition to aiding in getting a basic knowledge of the data, the EDA method also helps in revealing hidden patterns, anomalies, and possible research trajectories. This chapter will show how to efficiently load, visualize, and analyze data using Python tools like pandas, matplotlib, seaborn, and plotly. It will also explore statistical testing, hypothesis validation, and how crucial data preprocessing is for effective EDA.

4.2 LITERATURE REVIEW

The value of EDA in the data analysis process is well known, and it has been covered in great detail in the literature. John Tukey and Francis Anscombe, two well-known statisticians, were among the early proponents of the concept of visual data exploration, stressing the effectiveness of graphical tools in highlighting data patterns and outliers.

A wealth of modules specifically designed for data analysis and Python's user-friendly syntax have helped the programming language become extremely popular for EDA in the modern setting. Data scientists and analysts may easily manipulate and visualize data using the Python libraries, which include pandas, NumPy, matplotlib, and seaborn.

EDA in Python has been the subject of several publications, training programs, and books, underscoring its importance in the fields of data science and machine learning.

EDA in Python has been the subject of several publications, training programs, and books, underscoring its importance in the fields of data science and machine learning. EDA still serves as a crucial step in bridging the gap between unprocessed data

and useful insights as data-driven decision-making expands and changes across numerous industries. This chapter expands on the ideas introduced by those contributions and seeks to provide readers a useful and hands-on introduction to Python-based EDA.

4.3 EDA

An important element in the data analysis process that aids in understanding and making sense of our data is exploratory data analysis (EDA). We are able to find patterns, linkages, anomalies, and insights by visually and statistically analyzing the data. These findings help us to further analyze the data and make decisions. Python is a great option for EDA due of its robust ecosystem of libraries. We'll look at a variety of Python modules and methods in this chapter to do EDA efficiently.

4.4 CONSTRUCTING THE ENVIRONMENT

You need to set up your Python environment before we begin the EDA. The use of Jupyter Notebook or JupytersLab is advised because they offer a collaborative and interactive environment for data analysis.

Installing the following libraries for Python is necessary:

- **NumPy:** For manipulating arrays and performing numerical computations.
- **Pandas:** For analyzing and manipulating data.

For fundamental data visualization, use Matplotlib.

- **Seaborn:** For producing more sophisticated and appealing visualizations.
- **Plotly:** For online visualizations that are interactive.
- **Jupyter Widgets (ipywidgets):** To make your EDA notebooks more interactive.
- **Missingno:** For representing missing info visually.
- **SciPy:** used for statistical analysis and testing.

You can install these libraries using pip:

```
pip install numpy pandas matplotlib seaborn plotly ipywidgets missingno scip
```

4.5 UNDERSTANDING AND ADDING DATA

Data loading is the initial step in the EDA process. The default data import and processing tool in Python is the pandas package. Data can be loaded into SQL databases, CSV, Excel, and other formats. How to load a CSV file is as follows:

```
import pandas as pd

# Load data from a CSV file
data = pd.read_csv('your_data.csv')
```

Once you have your data loaded, start by understanding its basic characteristics:

```
# Display the first few rows of the dataset
data.head()

# Get a summary of the dataset
data.info()

# Descriptive statistics of numeric columns
data.describe()
```

4.6 DATA VISUALIZATION

Data visualization is an effective EDA technique. Python has a variety of libraries available for building both static and interactive charts. For this chapter, we'll concentrate on matplotlib, seaborn, and plotly.

4.7 BASIC MATPLOTLIB VISUALIZATION

Matplotlib is a flexible package that may be used to make a variety of static plots. Here's an easy illustration:

```
import matplotlib.pyplot as plt

# Plot a histogram of a numeric column
plt.hist(data['age'], bins=20)
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

4.8 SEABORN'S ENHANCED VISUALIZATION

Using a high-level interface, Seaborn, which is developed on top of Matplotlib, lets users create visually appealing plots:

```
import seaborn as sns

# Create a box plot to visualize the distribution of age
sns.boxplot(data=data, x='age')
plt.title('Age Distribution')
plt.show()
```

4.9 PLOTLY'S INTERACTIVE VISUALIZATION

Plotly is a fantastic tool for building interactive stories. It can smoothly connect with Jupyter Notebook:

```
import plotly.express as px

# Create an interactive histogram
fig = px.histogram(data, x='age', title='Age Distribution')
fig.show()
```

4.10 HANDLING MISSING DATA

Dealing with missing data is a critical aspect of EDA. Python provides the missingno library to visualize and handle missing values:

```
import missingno as msno

# Visualize missing data
msno.matrix(data)
```

4.11 ANALYZING RELATIONSHIPS

Understanding relationships between variables is key to gaining insights. You can use correlation matrices, scatter plots, and pair plots for this:

```
# Calculate the correlation matrix
correlation_matrix = data.corr()

# Create a heatmap to visualize the correlations
sns.heatmap(correlation_matrix, annot=True)
```

4.12 STATISTICAL ANALYSIS

Statistical analysis can provide valuable insights into your data. Python's scipy library is a great choice for conducting statistical tests:

```
from scipy.stats import ttest_ind

# Perform a t-test to compare two groups
group1 = data[data['group'] == 'A']['value']
group2 = data[data['group'] == 'B']['value']

t_stat, p_value = ttest_ind(group1, group2)
```

4.13 CONCLUSION

Exploratory Data Analysis (EDA) is a crucial stage of the data analysis process, and Python offers a robust toolkit to carry out EDA successfully. You can obtain insightful knowledge that directs your additional analysis and decision-making by visualizing data, dealing with missing values, and performing statistical analysis. Keep in mind that EDA is an iterative process that develops as you explore your data more

deeply; it is not a one-time effort. You are prepared to begin using Python for your EDA endeavors with the approaches and resources discussed in this chapter.

REFERENCES

1. *Tukey, John W. "Exploratory Data Analysis." Addison-Wesley, 1977.*
2. *John Tukey's seminal work, which introduced the concept of EDA and laid the foundation for modern data analysis techniques.*
3. *Anscombe, Francis J. "Graphs in Statistical Analysis." The American Statistician, 1973.*
4. *Francis Anscombe's paper that underscores the importance of visualizations and provides the famous Anscombe's Quartet as an example.*
5. *McKinney, Wes. "Python for Data Analysis." O'Reilly Media, 2017.*
6. *A comprehensive book that focuses on data analysis with Python, including a detailed section on EDA using pandas.*
7. *VanderPlas, Jake. "Python Data Science Handbook." O'Reilly Media, 2016.*
8. *This book covers various aspects of data science in Python, with a focus on EDA, data visualization, and analysis techniques.*
9. *Wickham, Hadley. "ggplot2: Elegant Graphics for Data Analysis." Springer, 2016.*
10. *While primarily focused on R, this book introduces the grammar of graphics and provides valuable insights into data visualization principles, which can be adapted to Python with libraries like seaborn.*
11. *Sahoo, K., Samal, A. K., Pramanik, J., & Pani, S. K. (2019). Exploratory data analysis using Python. International Journal of Innovative Technology and Exploring Engineering, 8(12), 4727-4735.*
12. *Samet, R., & Tural, S. (2010). Web based real-time meteorological data analysis and mapping information system. Proceedings of WSEAS Transactions of Information Science. and Applications, 1115-1125.*