Exploratory Data Analysis Basics – The Essential Guide

Rashmi Karan
Manager - Content

Updated on Nov 17, 2023 17:55 IST

Data science has become one of the fastest-growing fields with huge demands for skilled data scientists. Elements of data analysis like understanding and analyzing complex data, and visualizing it to extract conclusions and improve business decision-making are of the utmost importance now. Contributed by: Sooraj Bhupinder

Exploratory Data Analysis (EDA) has emerged as a popular approach to analyze data sets and present them visually. EDA has been successfully contributed towards giving maximum insight into the data set and data structures and has proved to be the most in-depth data analysis technique for data science projects. More and more data scientists are now mastering EDA and the market looks good for skilled professionals. EDA is nothing but a data exploration technique to understand the

various aspects of the data. It includes several techniques in a sequence that we have to follow. To know these techniques, you have to go through this article.

In this article, you will learn about Exploratory Data Analysis or EDA with Python on a small dataset for better understanding.

- What is Exploratory Data Analysis?
- Importance of EDA
- Goals of Exploratory Data Analysis
- Phases of EDA
- Types of Exploratory Data Analysis
- EDA Tools
- Steps involved in Exploratory Data Analysis.
- Detailed Exploratory Data Analysis with Python which will be done on a small data set.

So let us dive in.

What is Exploratory Data Analysis?

Exploratory Data Analysis (EDA) is a powerful approach to analyze data sets using summary statistics and graphical tools to gain insight into the data. EDA helps you to find anomalies like outliers or unusual observations in the data. It helps to identify patterns, understand possible relationships between variables, and generate interesting hypotheses using statistical methods. EDA is also helpful in cleaning data and representing the data graphically.

Importance of EDA

Exploratory Data Analysis is a data analytics process to understand the data in depth and to learn the different characteristics of data, often with visual means. This allows us to get a feel for our data better and find useful patterns in it.

The whole aim is to understand the data; understanding the data can be a lot of things when we are exploring the data. Few things we have to keep in mind while

exploring the data, we have to make sure that the data is clean and does not have redundancy or missing values, or even null values on the data set.

We must understand the variables through EDA. Along with that, we should be able to derive conclusions by gathering incites about the conclusive data interpretation to move on to a more complex process in the data processing life cycle.

*Also Read – **How to become a data scientist?***

Goals of EDA

EDA is an iterative process that helps to draw different actionable insights and devise data-based strategies. Typical goals of EDA are –

- Establishing variable distribution in your data set
- Creating a good-fitting model with no outliers to ensure no data quality problems
- Achieving a correct estimation for parameters
- Predicting the uncertainties of the estimates
- Drawing statistically significant conclusions
- Removing irregularities and unnecessary values from data
- Helping in preparing our dataset for analysis
- Enabling a machine learning model to predict our data set better
- Giving more accurate results
- Choosing a better machine-learning model

Why is EDA Crucial in Data Science?

EDA is a crucial process for data scientists to make any data-based prediction. It contributes towards –

- Spotting the obvious errors in the data sets
- Exposing trends, patterns, and relationships that are not evident
- Ensuring that the obtained results are valid and applicable to desired business outcomes

- Visualizing data through charts and graphs to present underlying information accurately
- Getting close to accurate answers about standard deviations, categorical variables, and confidence intervals
- Facilitating more sophisticated and accurate data analysis or modeling

Explore free data analytics courses

Phases of EDA

The phases of exploratory data analysis can be summarized in 7 steps –

- Know which problem area you will be covering and which questions you would answer
- Get a general idea of the dataset
- Define the types of data you have
- Choose the type of descriptive statistic
- Visualize the data
- Analyze the possible interactions between the variables of the dataset
- Draw some conclusions from all this analysis

Must Read – Data Scientist Salaries – Your Ultimate Guide

Types of Exploratory Data Analysis

There are four primary types of EDA:

1. **Univariate Non-Graphical EDA** – The data has only one variable and no relationships in univariate non-graphical EDA. This method is most commonly used to describe the data; make predictions of which population distribution(s) are compatible with the sample distribution, and find any existing patterns.

2. **Univariate Graphical EDA** – The graphical method summarizes the data visually, giving a complete picture of the data. Univariate Graphical EDA is further categorized into three types –

i. Histogram – Represents the total number of cases for particular values. A

histogram is mainly used for the graphical analysis of the univariate categorical data.

ii. Stem and Leaf plot – Represent all data values and the shape of the distribution.

iii. Box plots – Represent the five-number summary of minimum, first quartile median, third quartile, and maximum. Box plots can also be used as a visual tool to check normality or identify points that could be outliers

Explore free statistics for Data Science Online Courses

3. **Multivariate Non-Graphical EDA** – The multivariate non-graphical EDA depicts the relationship between two or more data variables using cross-tabulation or statistics.

4. **Multivariate Non-Graphical EDA** – Multivariate non-graphical EDA represents the relationship between two or more data sets. The most popular graphic is a bar plot or a bar chart. Here, every group represents one level of one of the variables, and every bar in a group represents levels of other variables. Some other multivariate graphics include:

I. Scatter plot – It plots data points on a horizontal and a vertical axis to depict the levels of the other variables and their dependencies.

II. Multivariate chart – It graphically represents the relationships between factors and response.

III. Bubble chart – It is a data visualization approach that represents multiple circles or bubbles in a 2D plot.

IV. Heat map -It graphically represents the data depicting values by colour.

Exploratory Data Analysis Tools

Some of the most common data science tools used to create an EDA include:

- **Python** – Python is an object-oriented programming language with high-level, built-in data structures. Other features like dynamic typing and dynamic binding work in favor of EDA. Python is extensively used to connect existing components and identify missing values in a data set.
- **Matplotlib –** Matplotlib is one of the most widely used in data science for all kinds of

graphics, such as bar charts, scatter charts, fever charts, and maps with Basemap, etc. Seaborn, another Python library based on Matplotlib, enables data scientists to create explanatory graphs from highly complex data.

- **R** – R is an open-source programming language in statistical computing and graphics. It has a wide range of applicability in statistical observations and data analysis.
- **ggplot2** – ggplot2 is a library that allows bar, point, line, area, maps, and scale charts. ggplot2 depends on other packages that need to be downloaded and installed.

Explore free Python courses	*Explore Data Science Courses*
Explore R programming certifications	*Explore programming courses*

Steps Involved in Exploratory Data Analysis

Exploratory Data Analysis follows a systematic set of steps to explore the data most efficiently.

1: Understand the Data: The very first and basic step is to understand the variables in the data set so you have to be pretty sure about what kind of data you are working on, what are the variables, like the number of columns and rows and how it looks like so that is your first step after loading the data into your program.

Note: Data collection is an important part of exploratory data analysis. It refers to the process of finding and loading data into our system. Good, reliable data can be found on various public sites or can be bought from private organizations

2: Clean the Data: The next step is to clean the data of redundancies; now, redundancies can be irregularities in the data; they can be some variables or some columns that are not necessary for making our conclusions or interpretations, so we can simply remove them; or they can be outliers, which can cause noise in the data or may overfit or underfit the model when you are also working on model building.

3: Analysis of Relationship Between Variables: Last, we must analyze the relationship between the variables.

Problem Statement

To perform EDA or Exploratory Data Analysis to know the student's score. We have to understand the problem that we are solving. We need to consider the entire dataset and the meaning of the variables.

Before performing EDA, let's have a look at the information provided in the Kegel dataset.

- There are a total of 1000 entries, ranging from 0 to 999.
- There are 8 columns in total.
- Each column has different data, i.e. gender, parental level of education, lunch, race/ethnicity, test preparation course, math score, writing score, and reading score.
- This dataset contains no null values.
- Through the values provided, we can check the math, reading, and writing scores using descriptions.

With the above information, we can carry out EDA.

Detailed Exploratory Data Analysis with Python

In this demo, we will use a small dataset to gather insights using exploratory data analysis.

Have a look at the screenshot below that shows EDA is performing on the data set taken from Kegel.

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns

In [3]: data = pd.read_csv("C:/Users/Waseem/Desktop/datasets/student.csv")
```

The very first thing you have to do is import certain libraries that you're going to need.

- Import pandas with an alias as pd
- Import numpy as np
- Import seaborn for visual representation, as we will use C-bond SNs to visualise the relationship between the variables

You can see the program has run successfully. Now, take the variable data and use the pandas library. The very first step is you have to import the data set and name of the data set i.e. students.csv

After you load the data into the program, you have to understand the data by understanding the variables inside your data. Check the image below.

```
In [5]: #1 understanding the data

In [6]: data.head()
```

Out[6]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

These scores are going to be important in our data set.

- While working on any model or making assumptions or conclusions like gender has to be there because it's decisive, it has to be male or female. This categorical value is going to be needed in our data set.
- There is an ethnicity that may be dropped. It's not necessarily very important in our data set.
- The parental level of education if we'll check for the unique value and we'll decide to firm idea on it. The next image shows the **tail** like the last two or five rows.

```
In [7]: data.tail()
```

Out[7]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

You can look at the values, and one thing you can ensure is it's starting from zero and going until 999. We can just say we have a thousand entries in this data set. Though it's not a very big data set, it is relatively not a small data set either. It's perfect for us because while doing the representation, it is going to be quite easy.

Now, let's check the **shape of the data** as well. We have a thousand rows and eight columns.

```
In [9]: data.shape
Out[9]: (1000, 8)
```

Let's just take a look at a few other key points. When we use the **describe** it only shows the math score, reading score, and writing score because all the other variables that we have are string objects; only the integer objects are shown.

As you can see, for all those values, 100 marks is the maximum, and for the minimum, we have a math score is zero, a reading score of 17, and a writing score of 10.

```
In [10]: data.describe()
```

Out[10]:

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

We can also check **columns and rows** separately.

```
In [12]: data.columns
Out[12]: Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
                'test preparation course', 'math score', 'reading score',
                'writing score'],
               dtype='object')
```

We can check for the **n unique** values, which is nothing but a function that returns a series with a number of distinct observations that were requested to be accessed. If we set the value of access to zero, then it finds the total number of unique observations over the index access.

```
In [16]: data.nunique()

Out[16]: gender                        2
         race/ethnicity                5
         parental level of education   6
         lunch                         2
         test preparation course       2
         math score                   81
         reading score                72
         writing score                77
         dtype: int64
```

If you want to check for any individual column, you can just write unique beside the value.

```
In [17]: data['gender'].unique()

Out[17]: array(['female', 'male'], dtype=object)
```

The first thing that will come to your mind is to check for the null values inside this dataset.

```
In [20]: #cleaning the data

In [21]: data.isnull().sum()

Out[21]: gender                        0
         race/ethnicity                0
         parental level of education   0
         lunch                         0
         test preparation course       0
         math score                    0
         reading score                 0
         writing score                 0
         dtype: int64
```

Inside this dataset, we have 0 null values; we don't have to worry about dropping any column just because there is no value or replacing it with some other values.

In some cases, some datasets are relatively very large if you have 7,000 or 8,000 values, and if you have even poopers in the null values or missing values inside those datasets, you have to be pretty sure about either if you want to leave those values untouched or if you want just to drop them or replace any value from them.

We'll be removing some values that we don't need, such as race/ethnicity and parental level of education.

```
In [27]: student = data.drop(['race/ethnicity', 'parental level of education'], axis=1)

In [28]: student.head()
```

Out[28]:

	gender	lunch	test preparation course	math score	reading score	writing score
0	female	standard	none	72	72	74
1	female	standard	completed	69	90	88
2	female	standard	none	90	95	93
3	male	free/reduced	none	47	57	44
4	male	standard	none	76	78	75

Now, moving on to the relationship analysis.

```
In [30]: #3 realtionship analysis
```

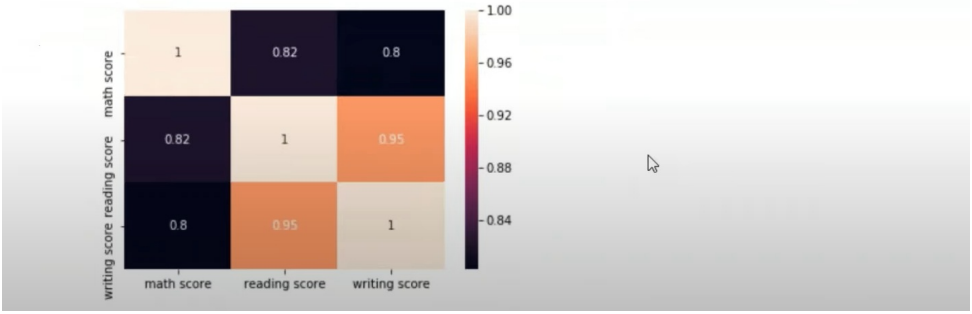
Our correlation matrix gives us a wider X perspective on what we are dealing with here.

A **correlation matrix** is a table showing the correlation coefficient between variables and each cell in the table. The correlation between two variables and the correlation matrix is used to summarize data as input into a more advanced analysis and diagnostic for an advanced analysis.

```
In [32]: corelation = student.corr()
```

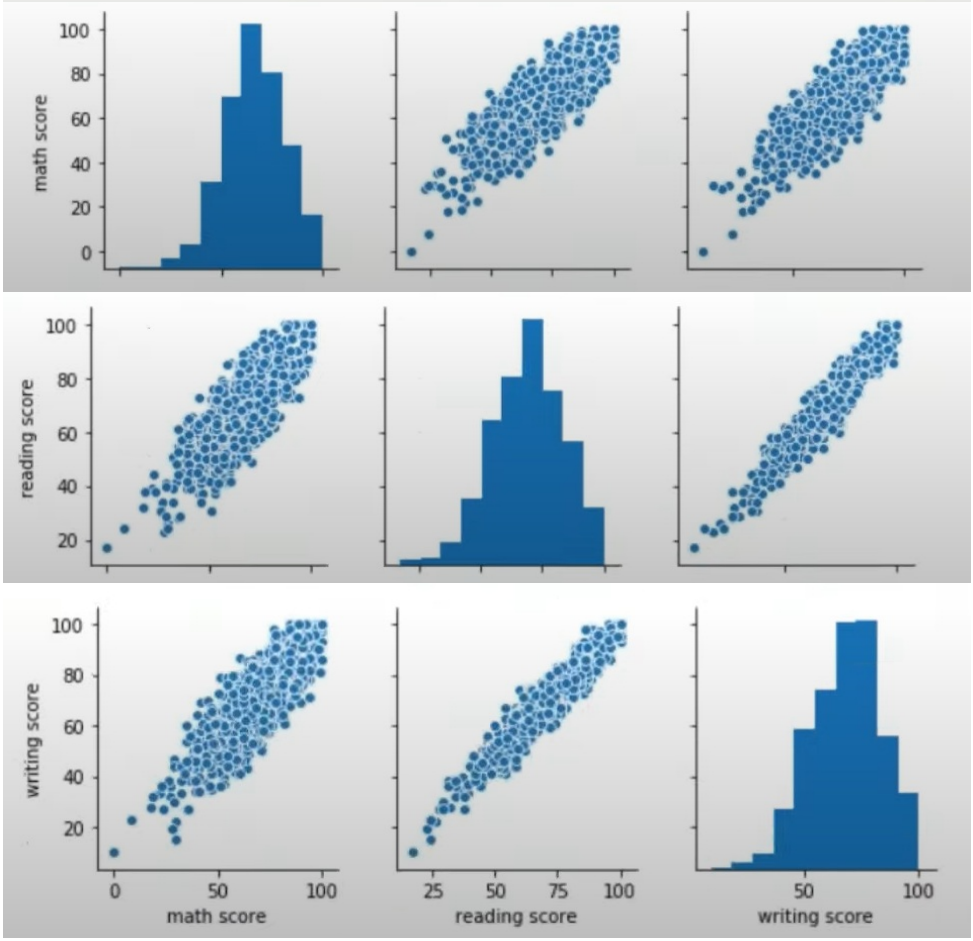
Take the data and correlation value and see whether there is any error; if not then put it inside a **heat map base** using the SNS library or C bond library. SNS is the alias used for inputting the library.

```
In [36]: sns.heatmap(corelation, xticklabels=corelation.columns, yticklabels=corelation.columns
                    ,annot=True)
Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x156b89f5dd8>
```



Now, we are going to plot a few other plots. There is one pair plot that we can actually lord. A **pair plot** helps you to visualize the relationship between two variables where the variables can be continuously categorical; a pair plot is usually a grid of slots for each variable in your data.

```
In [*]: sns.pairplot(student)
Out[37]: <seaborn.axisgrid.PairGrid at 0x156be6bfb70>
```

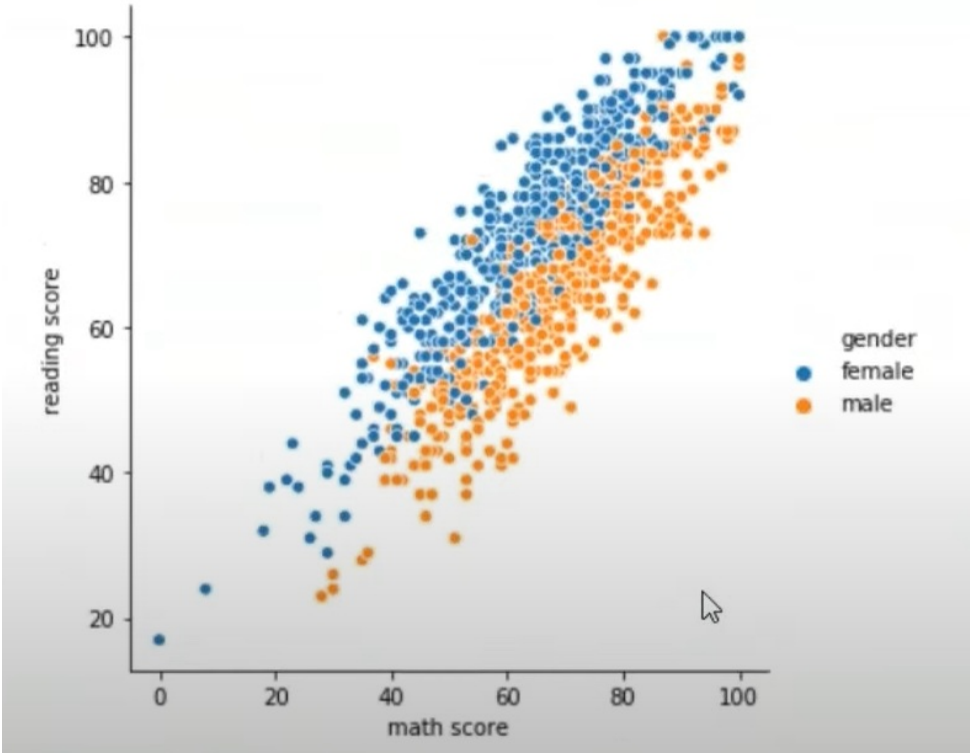


As you can see in the above images, we have math score, reading score, and writing score. These are quite descriptive when you look at them, all are in increasing firmness. This is also not quite decisive when we are looking at the conclusion.

Now, we move on to **scatter plot**, he scatter plot is a type of data display that

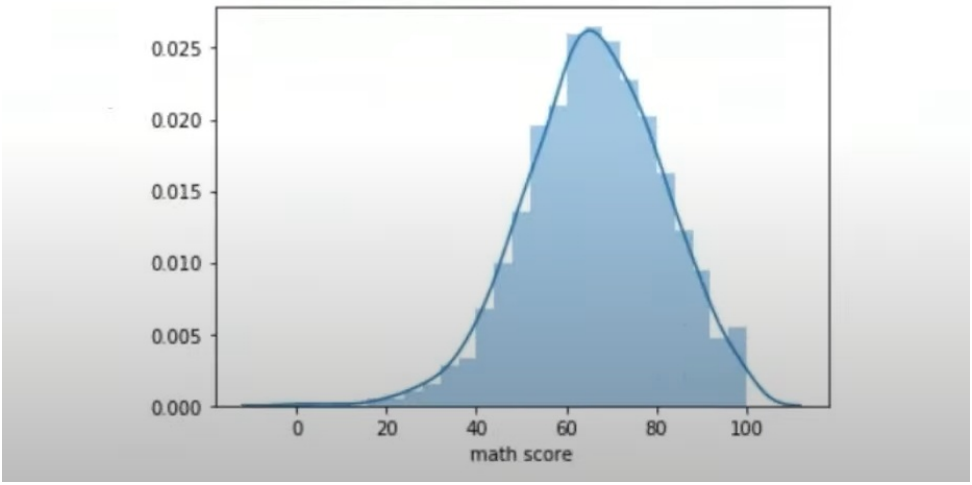
shows the relationship between two numerical variables by plotting each member of the data cell as a point whose left and right parenthesis coordinates relate values for two variables.

```
In [38]: sns.relplot(x= 'math score', y='reading score', hue='gender', data=student)
Out[38]: <seaborn.axisgrid.FacetGrid at 0x156bef75940>
```



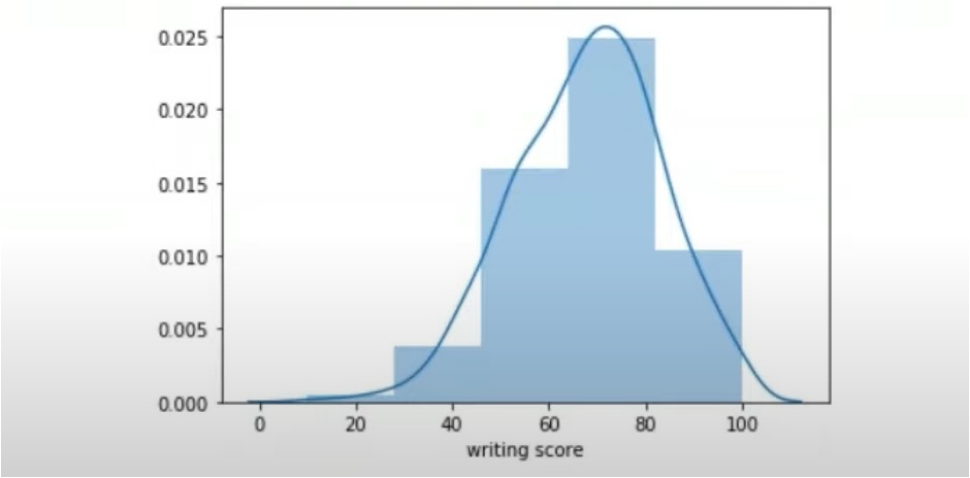
Moving on to the next plot i.e. **histogram**. A histogram is a graphical display of data using powers of different heights. In the histogram, each bar groups numbers into ranges, the taller bar shows more data range. For this, we'll be using the SNS **distribution plot**.

```
In [44]: sns.distplot(student['math score'])
Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x156bebbb198>
```



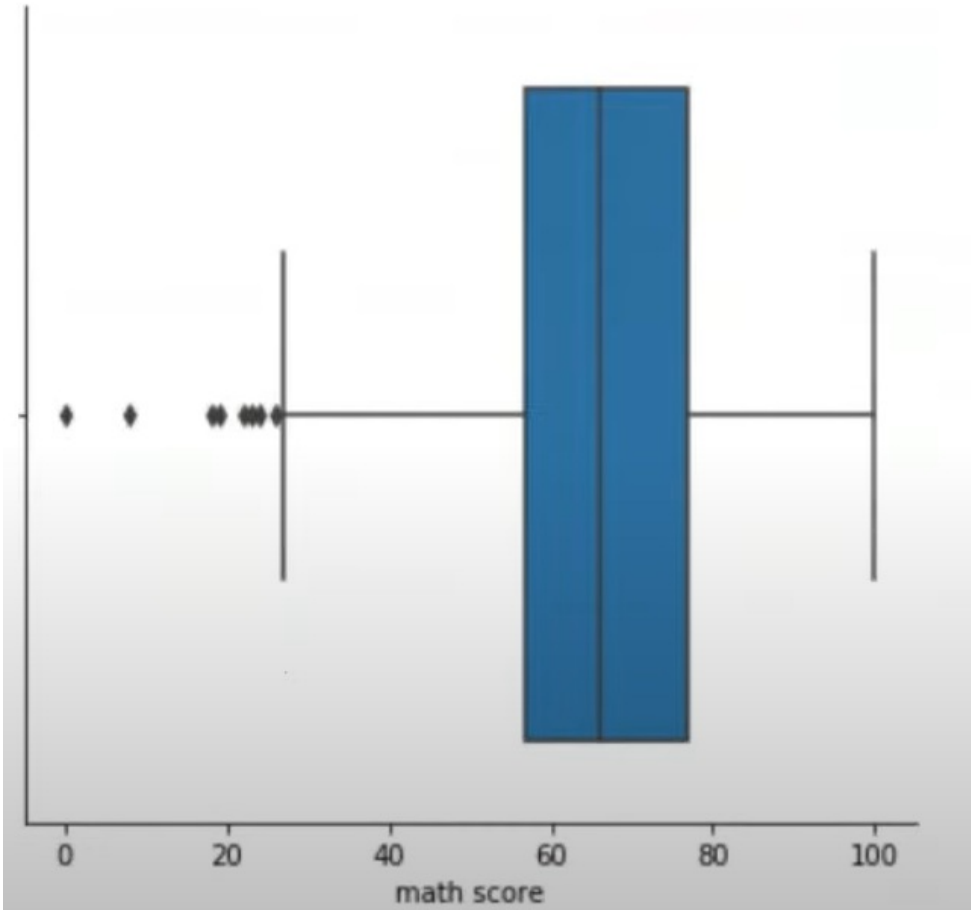
We can check all the values from the histogram to analyze the relationship between datasets and can add **bins** as well.

```
In [48]: sns.distplot(student['writing score'], bins=5)
Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x156c0750d30>
```



You can check the value of the variables by using the **Categorical plot**.

```
In [51]: sns.catplot(x='math score', kind= 'box', data= student)
Out[51]: <seaborn.axisgrid.FacetGrid at 0x156c0750160>
```



We can conclude a few things that we have looked into our data. We can conclude one thing over another for each step over here.

This is how you do EDA on any data. It is relatively small data with only 8 columns and a thousand values. Sometimes a few datasets, like if you're working on stock prediction, at least 18,19 or 200 columns have entries in the dataset.

Conclusion

Exploratory Data Analysis is a crucial way to understand the data you will be working

on and is a highly recommended method for a correct research methodology. EDA helps to explore, describe, summarize and visualize the data collected in the random variables of the project or research of interest through the application of simple data summary techniques and graphic methods without assuming assumptions for their interpretation. Data scientists have been using EDA to reflect the data and variables accurately.

FAQs - EDA

Why is EDA important in data analysis?

How do I start EDA for a new dataset?

What is the purpose of data visualization in EDA?

Can EDA identify outliers and anomalies in data?

Are there software tools specifically designed for EDA?