

✦ Member-only story

Data Visualization for Exploratory Data Analysis (EDA) in Python

Python Data Visualization Guide



Gözde Madendere · [Follow](#)

6 min read · May 18, 2024



119



Data visualization is an essential part of Exploratory Data Analysis, as it helps to analyze and visualize the data to gain enlightening insights into its distribution, relationships between variables, and potential outliers.



Photo from [Pexels](#)

Python has a rich set of libraries that enable us to create visualizations quickly and efficiently. Several types of visualizations are commonly used in EDA using Python, including:

- **Bar charts:** Used to show comparisons between different categories.
- **Line charts:** Used to show trends over time or across different categories.
- **Pie charts:** Used to show proportions or percentages of different categories.
- **Histograms:** Used to show the distribution of a single variable.
- **Heatmaps:** Used to show the correlation between different variables.
- **Scatter plots:** Used to show the relationship between two continuous variables.

- **Box plots:** Used to show the distribution of a variable and identify outliers.

The Steps for Creating Data Visualizations Using Python

1. **Understanding the business problem:** This first step is important as we will be able to focus on getting the right visualizations.
2. **Importing necessary libraries:** Import the necessary libraries such as Pandas, Seaborn, Matplotlib, Plotly.
3. **Loading the dataset:** Load the dataset that you want to visualize.
4. **Data cleaning and preprocessing:** Clean and preprocess the data by removing missing values, duplicates, and outliers. Also, convert categorical data into numerical data.
5. **Statistics summary:** Calculate descriptive statistics such as mean, median, mode, standard deviation, and correlation coefficients to understand the relationships between variables.
6. **Data visualization & Interpretation:** Create visualizations to understand the distribution, relationships, and patterns in the data. After that interpret the visualizations to gain enlightening insights and conclusions about the data.

1. Understanding the business problem

Cardiovascular diseases are the leading cause of death globally. According to the WHO, an estimated 17.9 million people died from heart disease each year. 85% of these deaths are caused by heart attack and stroke.

In this article, we will explore the Heart Attack dataset from Kaggle and use Python to create data visualizations for EDA.

The dataset contains data on patients with various variables such as age, gender, blood pressure, cholesterol level, and whether or not they had a heart attack. The goal of this dataset is to predict whether a patient is at risk of having a heart attack based on their medical attributes.

2. Importing necessary libraries

```
# import libraries
import pandas as pd
import numpy as np

# data visualization
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

3. Loading the dataset

Let's load the data into a Pandas DataFrame and start exploring it.

```
heart = pd.read_csv('heart.csv')
```

Now that we have loaded the data, let's take a look at the first few rows of the DataFrame to get an idea of what the data looks like.

```
heart.head()
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	Higher Risk
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	Higher Risk
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	Higher Risk
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	Higher Risk
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	Higher Risk

We can see that the dataset contains 14 columns, including the target column (output), which indicates whether or not the patient had a heart attack. Now let's start creating visualizations.

4. Data cleaning and preprocessing

The purpose of data cleaning is to get our data ready to analyze and visualize.

```
# check if there are any Null values
heart.isnull().sum().sort_values(ascending=False).head(11)
```

```
age      0
sex      0
cp       0
trtbps   0
chol     0
fbs      0
restecg  0
thalachh 0
exng     0
oldpeak  0
slp      0
dtype: int64
```

As we can see here, there are no missing values in this case

```
# check duplicated values
heart.duplicated().sum()
```

1

```
# drop duplicated values
heart.drop_duplicates(keep='first', inplace=True)
```

0

Now our data is clean.

5. Statistics summary

```
# getting the statistical summary of dataset
heart.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	302.0	54.420530	9.047970	29.0	48.00	55.5	61.00	77.0
sex	302.0	0.682119	0.466426	0.0	0.00	1.0	1.00	1.0
cp	302.0	0.963576	1.032044	0.0	0.00	1.0	2.00	3.0
trtbps	302.0	131.602649	17.563394	94.0	120.00	130.0	140.00	200.0
chol	302.0	246.500000	51.753489	126.0	211.00	240.5	274.75	564.0
fbs	302.0	0.149007	0.356686	0.0	0.00	0.0	0.00	1.0
restecg	302.0	0.526490	0.526027	0.0	0.00	1.0	1.00	2.0
thalachh	302.0	149.569536	22.903527	71.0	133.25	152.5	166.00	202.0
exng	302.0	0.327815	0.470196	0.0	0.00	0.0	1.00	1.0
oldpeak	302.0	1.043046	1.161452	0.0	0.00	0.8	1.60	6.2
slp	302.0	1.397351	0.616274	0.0	1.00	1.0	2.00	2.0
caa	302.0	0.718543	1.006748	0.0	0.00	0.0	1.00	4.0
thall	302.0	2.314570	0.613026	0.0	2.00	2.0	3.00	3.0
output	302.0	0.543046	0.498970	0.0	0.00	1.0	1.00	1.0

The main inference that we can get here is, for most of the columns, the mean value is similar to the median value (50th percentile: 50%).

6. Data visualization & Interpretation

- Gender-based data visualization

```
# Compare Heart Attack vs Sex
df = pd.crosstab(heart['output'], heart['sex'])
sns.set_style("white")

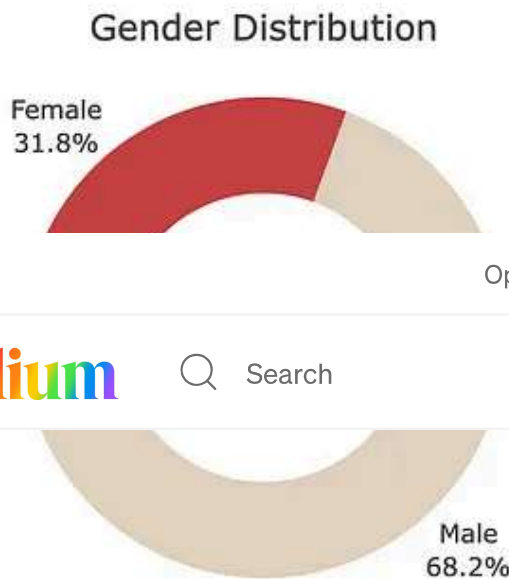
df.plot(kind="bar",
        figsize=(6,6),
        color=['#c64343', '#e1d3c1']);

plt.title("Heart Attack Risk vs Sex ", fontsize=16)
plt.xlabel("0 = Lower Risk          1 = Higher Risk", fontsize=16)
plt.ylabel("Amount", fontsize=16)
```



```
text="<span style='font-size: 26px; color=#555; font-fam
```

```
fig.update_traces(textposition='outside', textinfo='percent+label', rotation=20  
fig.show()
```



[Open in app](#) ↗

Medium

Search

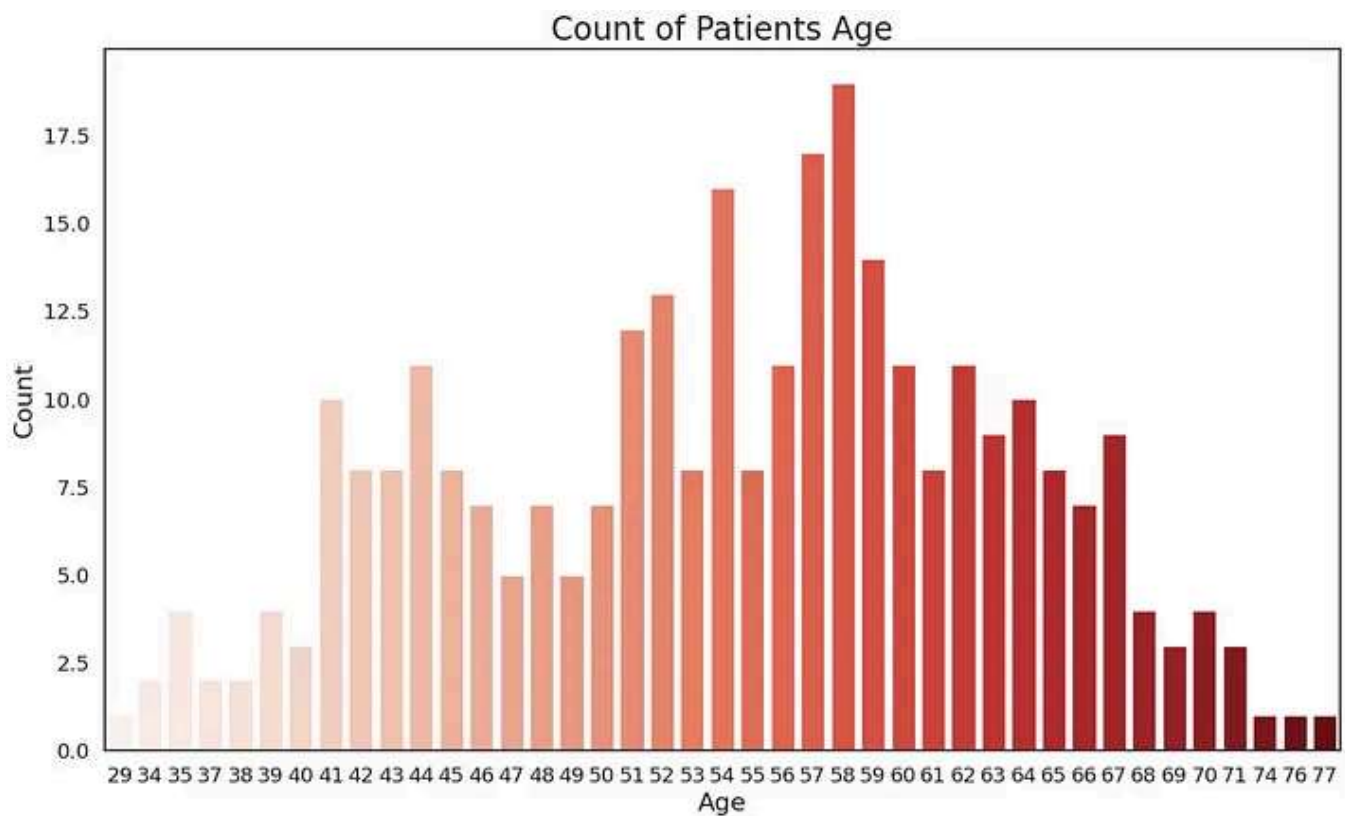
Write



Interpretation: Males have a higher risk of heart attack.

- **Age-based data visualization**

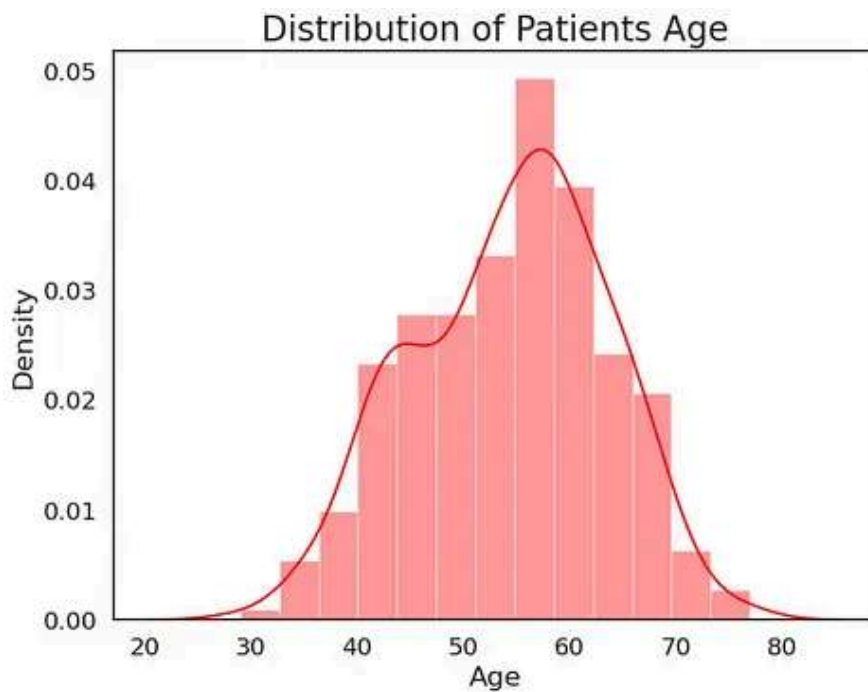
```
plt.figure(figsize=(14,8))  
sns.set(font_scale=1.2)  
sns.set_style("white")  
  
sns.countplot(x=heart["age"],  
              palette='Reds')  
  
plt.title("Count of Patients Age", fontsize=20)  
plt.xlabel("Age", fontsize=16)  
plt.ylabel("Count", fontsize=16)  
plt.show()
```



```
# age based analysis
sns.set(font_scale=1.3)

plt.figure(figsize=(8,6))
sns.set_style("white")
sns.distplot(heart['age'],
             color='red',
             kde=True)

plt.title("Distribution of Patients Age",fontsize=20)
plt.xlabel("Age",fontsize=16)
plt.ylabel("Density",fontsize=16)
plt.show()
```



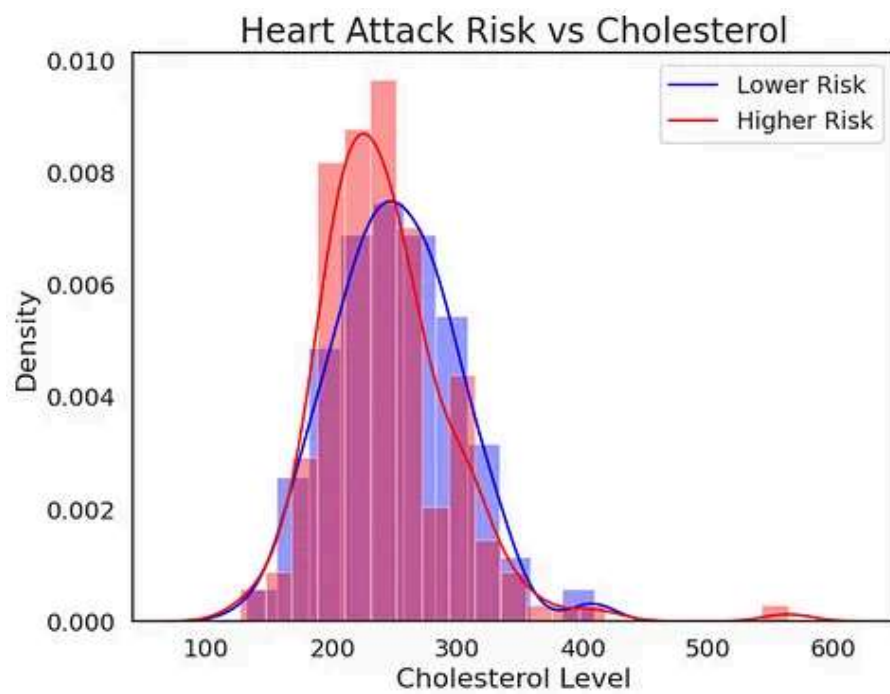
Interpretation: Most of the patients have age (50–60). In which the maximum number of Patients have age 58.

- **Cholesterol level-based data visualization**

```
# Attack vs Cholesterol analysis
sns.set(font_scale=1.3)
plt.figure(figsize=(8,6))
sns.set_style("white")

sns.distplot(heart[heart["output"]==0]["chol"],
             color="blue")
sns.distplot(heart[heart["output"]==1]["chol"],
             color="red")

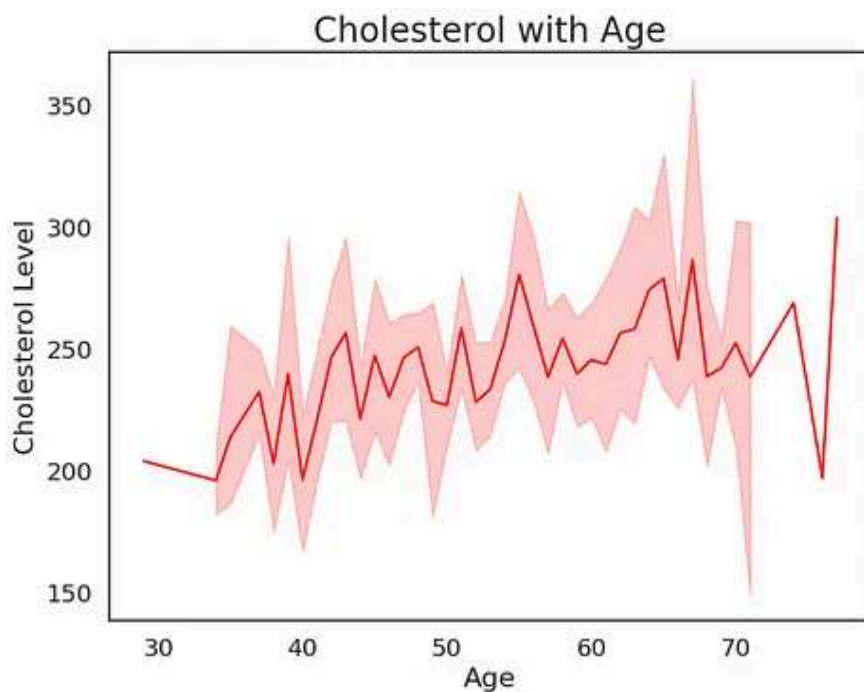
plt.title("Heart Attack Risk vs Cholesterol", size=20)
plt.xlabel("Cholesterol Level", fontsize=16)
plt.ylabel("Density", fontsize=16)
plt.legend(["Lower Risk", "Higher Risk"], fontsize=14)
plt.show()
```



```
plt.figure(figsize=(8,6))

sns.lineplot(y="chol",
             x="age",
             data=heart,
             color="red")

plt.title("Cholesterol with Age",fontsize=20)
plt.xlabel("Age",fontsize=16)
plt.ylabel("Cholesterol Level",fontsize=16)
plt.show()
```



Interpretations:

- Most of the patients have cholesterol levels between 200–300.
- There is a high probability of increase in the Level of Cholesterol in the body with age.
- **Chest Pain Type-based data visualization**

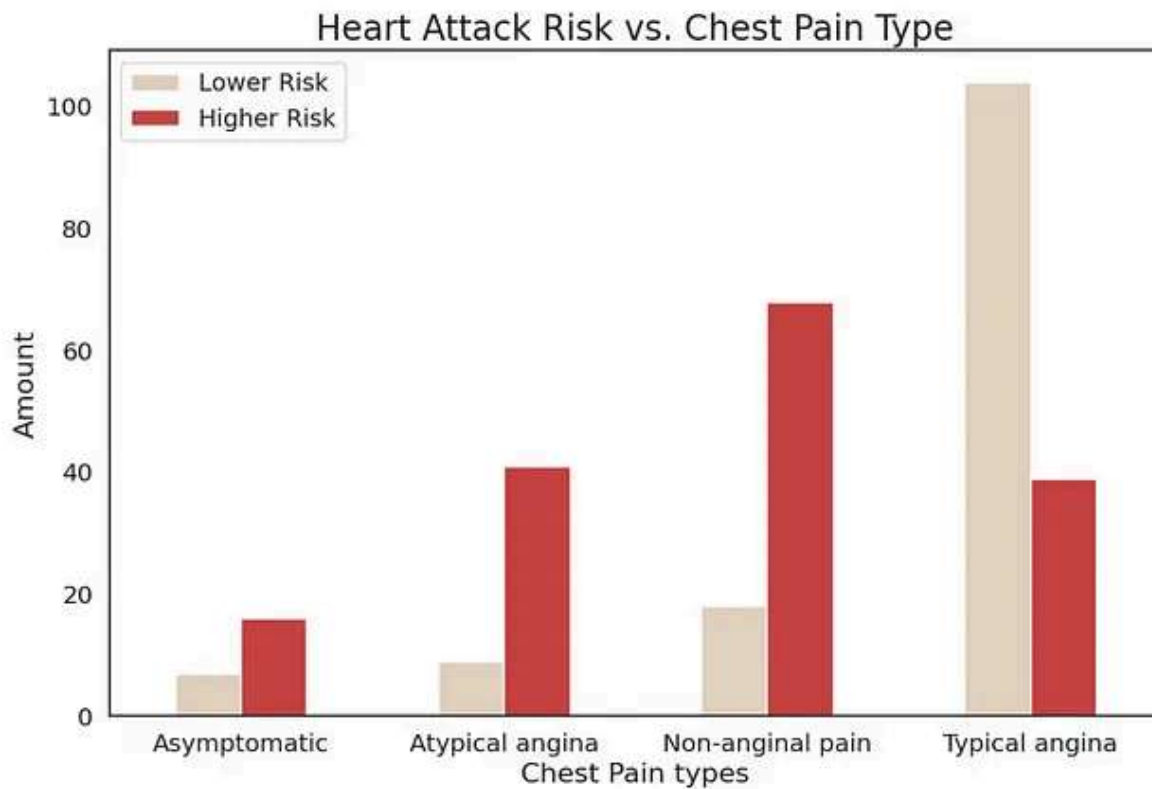
```
# Relation of Heart Attack with Chest Pain Type
df = pd.crosstab(heart3['cp'], heart['output'])

# Make the crosstab more visual
sns.set(font_scale=1.3)
sns.set_style("white")

df.plot(kind='bar',
        figsize=(11,7),
        color=['#e1d3c1', '#c64343']);

plt.title("Heart Attack Risk vs. Chest Pain Type", fontsize=20)
plt.xlabel("Chest Pain types", fontsize=16)
plt.ylabel("Amount", fontsize=16)
```

```
plt.legend(['Lower Risk', 'Higher Risk'], fontsize=14)
plt.xticks(rotation=0);
```



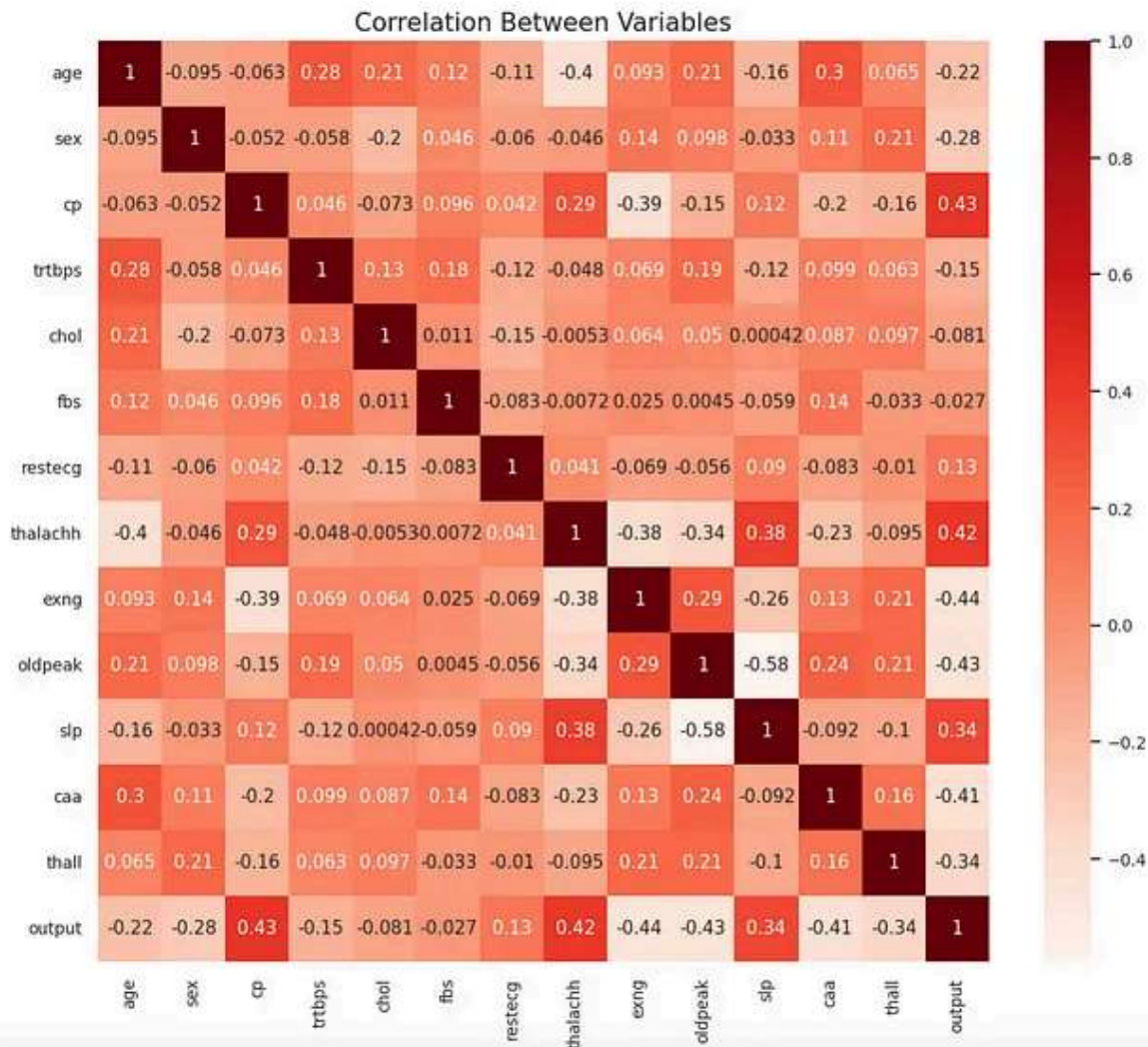
Interpretations:

- Most of the patients have the type of **Typical Angina**.
- Patients with **Non-anginal pain** have a higher risk of heart attack.
- **Correlation-based data visualization**

```
plt.figure(figsize=(12,10))
sns.set(font_scale=0.9)

sns.heatmap(heart.corr(),
            annot=True,
            cmap='Reds')
```

```
plt.title("Correlation Between Variables", size=15)
plt.show()
```



Interpretations:

The heatmap shows us there is a correlation between

- Chest Pain Type (cp) and Output
- Maximum heart rate achieved (thalachh) and Output
- Slope (slp) and Output

We can also see that there is a weak correlation between

- oldpeak: Previous peak and output
- caa: Number of major vessels and output
- exng: Exercise-induced angina

Conclusion

In this post, we examined our dataset using data visualizations creating several plots like bar chart, pie chart, line chart, histogram, heatmap.

The main purpose of EDA and data visualization are to help understand data before making any assumptions. They helps us to see distribution, summary statistics, relationships between variables and outliers.

Thank you for reading!

Are you interested in Data Science? Let's connect on [Linkedin](#).

Data Visualization

Python

Data Science

Data Scientist

Data Science Training



Written by Gözde Madendere

526 Followers

Data Scientist | Data Analyst | BS in Statistics

Follow



More from Gözde Madendere



Gözde Madendere in Towards Dev

9 Advanced SQL Queries for Data Mastery

Unlocking the Power of SQL

🌟 • 2 min read • Jan 8, 2024



177



Gözde Madendere in DataDrivenInvestor

Selecting and Filtering Data in Python Pandas

loc, iloc, isin, between and string methods

4 min read • Feb 2, 2024



180





Gözde Madendere in Towards Dev

Top 10 Advanced SQL Queries

SQL (Structured Query Language) is a versatile tool for managing and querying dat...

🌟 · 3 min read · Nov 24, 2023



135



3



Gözde Madendere in Towards Dev

SQL CASE Statement with Code Examples

In the world of SQL, the CASE statement stands as a versatile tool, allowing developer...

🌟 · 3 min read · May 5, 2024



102



See all from Gözde Madendere

Recommended from Medium



 Sithi Asma Basheer

Advanced SQL Queries you must know

The Top 15 Advanced SQL commands for Data Analysis and Data Engineering!

🌟 · 9 min read · May 22, 2024



Lists



Predictive Modeling w/ Python

20 stories · 1288 saves



Practical Guides to Machine Learning

10 stories · 1548 saves



Coding & Development

11 stories · 652 saves



ChatGPT prompts

48 stories · 1671 saves




 Benedict Neo in bitgrit Data Science Publication

How to Setup a Windows Laptop for Data Science

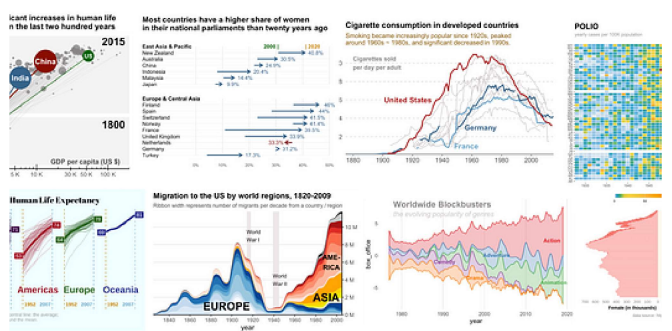
powershell, WSL, Python, Cuda and more!

5 min read · Jun 4, 2024

 Tristen Wallace

Data Wrangling 101: Preparing Your Data in Jupyter

8 min read · Jan 4, 2024



 Bo Yuan, Ph.D.

Awesome Strategies to Visualize Change with Time

This article summarizes effective strategies to visualize temporal changes, illustrated with...

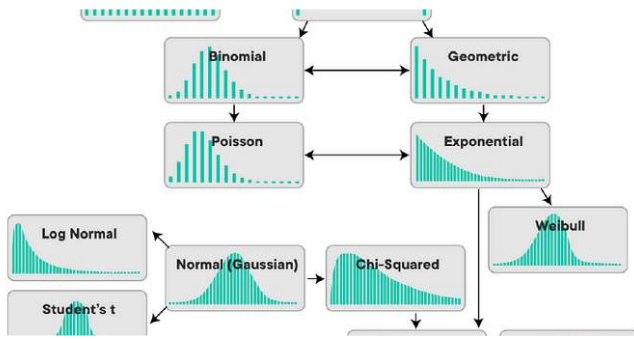
5 min read · May 8, 2024

👏 343 💬 2

🔖
...

👏 1.1K 💬 12

🔖
...



 Sachinsoni

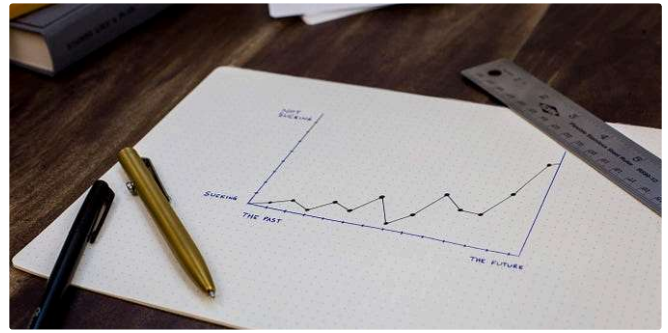
Understanding Different Kinds of Distributions in Statistics

Statistics is a powerful tool for making sense of data, and at its core lies the concept of...

8 min read · Apr 26, 2024

👏 346 💬 6

🔖
...



 Muhammad Ardi in Python in Plain English

Data Visualization with Matplotlib and Seaborn (Part 1/5)

A guide to creating stunning graphs in Python: Part 1—drawing lines, curves, and...

15 min read · Jan 2, 2024

👏 85 💬

🔖
...

See more recommendations