رواد مصر الرقمية

وزارة الاتصالات
وتكنولوجيا المعلومات

Introduction to Data Analysis

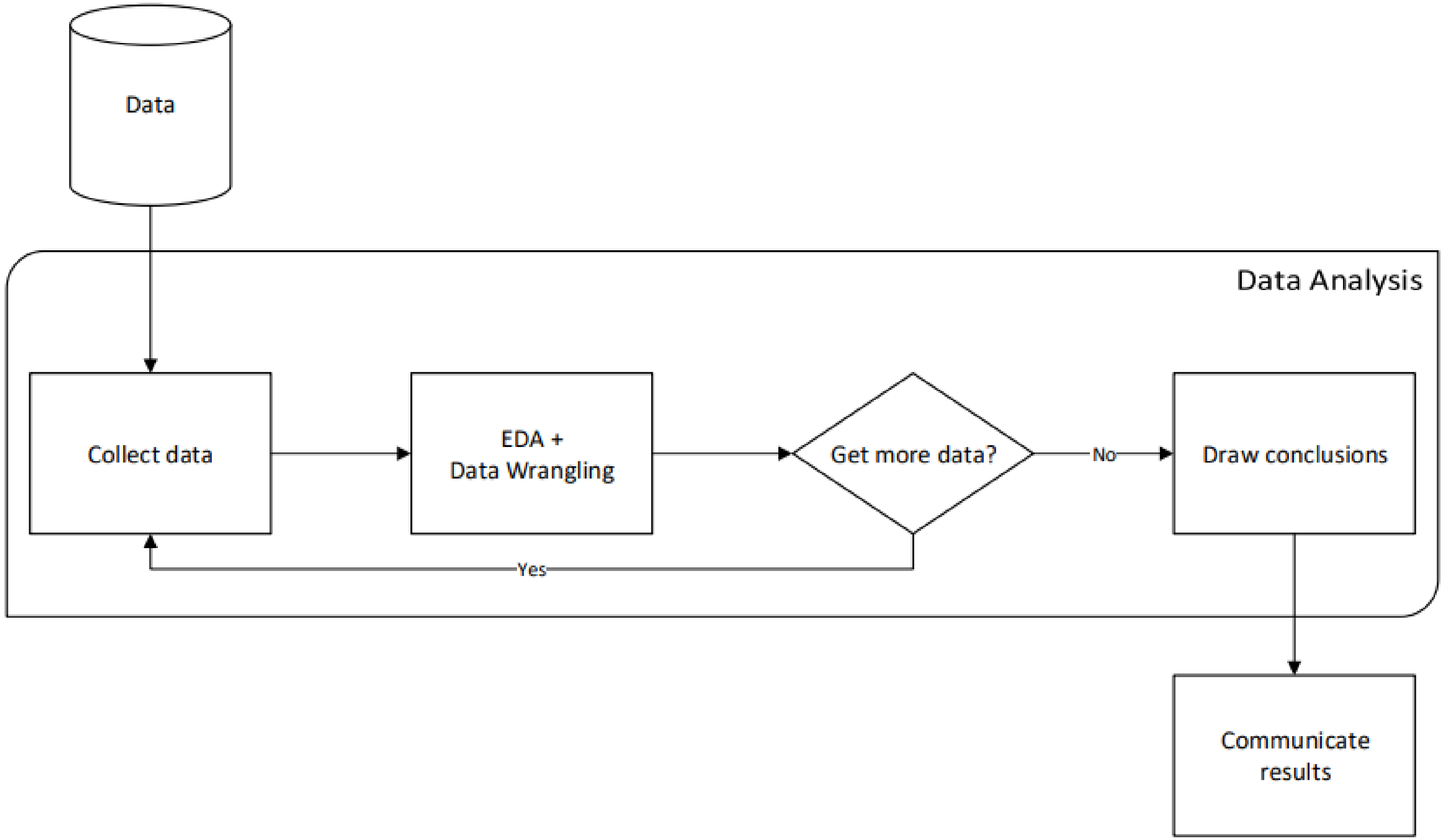
07/07/2024

Course Overview

Analysing data with Python is an essential skill for Data Scientists and Data Analysts. This course will take you from the basics of data analysis with Python to building and evaluating data models.

The Fundamentals of Data Analysis

Data analysis is a highly iterative process involving **collection**, **preparation** (wrangling), exploratory data analysis (**EDA**), and drawing **conclusions**. The following diagram depicts a generalized workflow:



Data Collection

Data collection is the natural first step for any data analysis—we can't analyze data we don't have!

While data can come from anywhere, we will explore the following sources throughout this course:

- Web scraping to extract data from a website's HTML (often with Python packages such as selenium, requests, scrapy, and beautifulsoup)
- Application programming interfaces (APIs) for web services from which we can collect data with HTTP requests (perhaps using cURL or the requests Python package)
- Databases (data can be extracted with SQL or another database-querying language)
- Internet resources that provide data for download, such as government websites
- Log files

Data Wrangling

Data wrangling is the process of preparing the data and getting it into a format that can be used for analysis.

The following are some issues we may encounter with our data:

- **Human errors**: Data is recorded (or even collected) incorrectly, such as putting 100 instead of 1000, or typos. In addition, there may be multiple versions of the same entry recorded, such as New York City, NYC, and nyc.
- **Computer error**: Perhaps we weren't recording entries for a while (missing data).
- **Unexpected values**: Maybe whoever was recording the data decided to use a question mark for a missing value in a numeric column, so now all the entries in the column will be treated as text instead of numeric values.
- **Incomplete information**: Think of a survey with optional questions; not everyone will answer them, so we will have missing data, but not due to computer or human error.

Data Wrangling

Data wrangling is the process of preparing the data and getting it into a format that can be used for analysis.

The following are some issues we may encounter with our data:

- **Resolution**: The data may have been collected per second, while we need hourly data for our analysis.
- **Relevance of the fields**: Often, data is collected or generated as a product of some process rather than explicitly for our analysis. In order to get it to a usable state, we will have to clean it up.
- **Format of the data**: Data may be recorded in a format that isn't conducive to analysis, which will require us to reshape it.
- **Misconfigurations in the data-recording process**: Data coming from sources such as misconfigured trackers and/or webhooks may be missing fields or passed in the wrong order.

Exploratory Data Analysis

EDA is a phenomenon under data analysis used for gaining a better understanding of data aspects like:

- Main features of data
- Variables and relationships that hold between them
- Identifying which variables are important for our problem

During EDA, we use visualizations and summary statistics to get a better understanding of the data. Since the human brain excels at picking out visual patterns, data visualization is essential to any analysis.

Depending on our data, we may create plots to see how a variable of interest has evolved over time, compare how many observations belong to each category, find outliers, look at distributions of continuous and discrete variables, and much more.

Data Types

When calculating summary statistics, we must keep the type of data we collected in mind.

Data can be:

- Quantitative (measurable quantities)
  - **Interval Scale**: Equal intervals between values. No true zero point.
  - **Ratio Scale**: Equal intervals between values. True zero point.

- Categorical (descriptions, groupings, or categories)
  - **Nominal**: Assign numeric value to each category (e.g., on = 1, off = 0).The numeric order is arbitrary.
  - **Ordinal**: Categories have a ranking order (e.g., low < medium < high).

Drawing Conclusions

After we have collected the data for our analysis, cleaned it up, and performed some thorough EDA, it is time to draw conclusions. This is where we summarize our findings from EDA and decide the next steps:

- Did we notice any patterns or relationships when visualizing the data?
- Does it look like we can make accurate predictions from our data? Does it make sense to move to modeling the data?
- Should we handle missing data points? How?
- How is the data distributed?
- Does the data help us answer the questions we have or give insight into the problem we are investigating?
- Do we need to collect new or additional data?

رواد مصر الرقمية

Statistical Foundations

وزارة الاتصالات وتكنولوجيا المعلومات

When we want to make observations about the data we are analyzing, we often, if not always, turn to **statistics** in some fashion.

The data we have is referred to as the **sample**, which was observed from (and is a subset of) the **population**. Two broad categories of statistics are:

- **Descriptive statistics**, as the name implies, we are looking to describe the sample.
- **Inferential statistics** involves using the sample statistics to infer, or deduce, something about the population, such as the underlying distribution.

Sampling

There's an important thing to remember before we attempt any analysis: our sample must be a **random sample** that is representative of the population.

This means that the data must be sampled without bias and that we should have (ideally) members of all distinct groups from the population in our sample.

For example, if we are asking people whether they like a certain sports team, we can't only ask fans of the team.

Descriptive Statistics

Descriptive statistics are used to describe and/or summarize the data we are working with.

We will begin our discussion of descriptive statistics with **univariate statistics**; univariate simply means that these statistics are calculated from one (**uni**) variable.

We can start our summarization of the data with a measure of **central tendency**, which describes where most of the data is centered around, and a measure of **spread** or **dispersion**, which indicates how far apart values are.

Measures of Central Tendency

Measures of central tendency describe the center of our distribution of data. There are three common statistics that are used as measures of center: **mean**, **median**, and **mode**.

Mean

The sample mean is calculated by summing all the values and dividing by the count of values; for example, the mean of the numbers 0, 1, 1, 2, and 9 is 2.6 ((0 + 1 + 1 + 2 + 9)/5):

$$\bar{x} = \frac{\sum_1^n x_i}{n}$$

One important thing to note about the mean is that it is very sensitive to **outliers** (values created by a different generative process than our distribution).

Measures of Central Tendency

Median

Unlike the mean, the median is robust to outliers because it is the 50th percentile of our data; this means that 50% of the values are greater than the median and 50% are less than the median.

The median is calculated by taking the middle value from an **ordered** list of values; in cases where we have an even number of values, we take the mean of the middle two values.

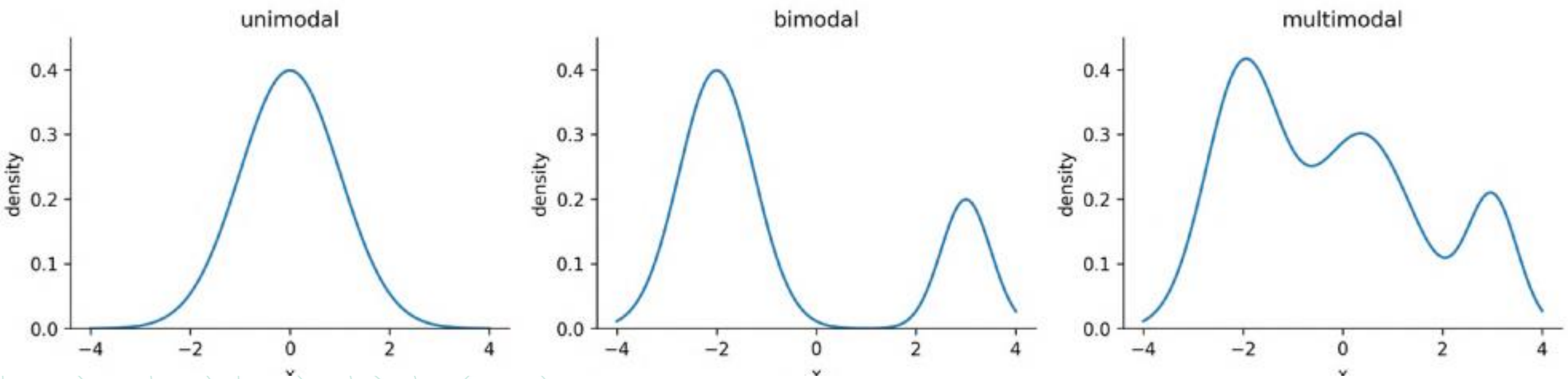
If we take the numbers 0, 1, 1, 2, and 9 again, our median is 1.

Measures of Central Tendency

Mode

The mode is the most common value in the data (if we, once again, have the numbers 0, 1, 1, 2, and 9, then 1 is the mode).

In practice, we will often hear things such as the distribution is *bimodal* or *multimodal* (as opposed to *unimodal*) in cases where the distribution has two or more most popular values.



Most of the time when we're describing our **continuous data**, we will use either the **mean** or the **median** as our measure of central tendency. When working with **categorical data** we will typically use the **mode**.

Measures of Spread

Knowing where the center of the distribution is only gets us partially to being able to summarize the distribution of our data—we need to know how values fall around the center and how far apart they are.

We have several ways to describe the spread of a distribution, and which one we choose will depend on the situation and the data.

Range

The range is the distance between the smallest value (minimum) and the largest value (maximum).

$$range = \max(X) - \min(X)$$

While the range provides upper and lower bounds for our data, it isn't always the best measure of spread. **Outliers** can make the range ineffective, as they can skew the true variability of the data.

The range only shows overall data dispersion, **not** how data is dispersed around the center. This is why we use variance.

Measures of Spread

Variance

The variance describes how far apart observations are spread out from their average value (the mean). It is calculated as the average squared distance from the mean.

$$s^2 = \frac{\sum_1^n (x_i - \bar{x})^2}{n-1}$$

Note that the distances must be squared so that distances below the mean don’t cancel out those above the mean.

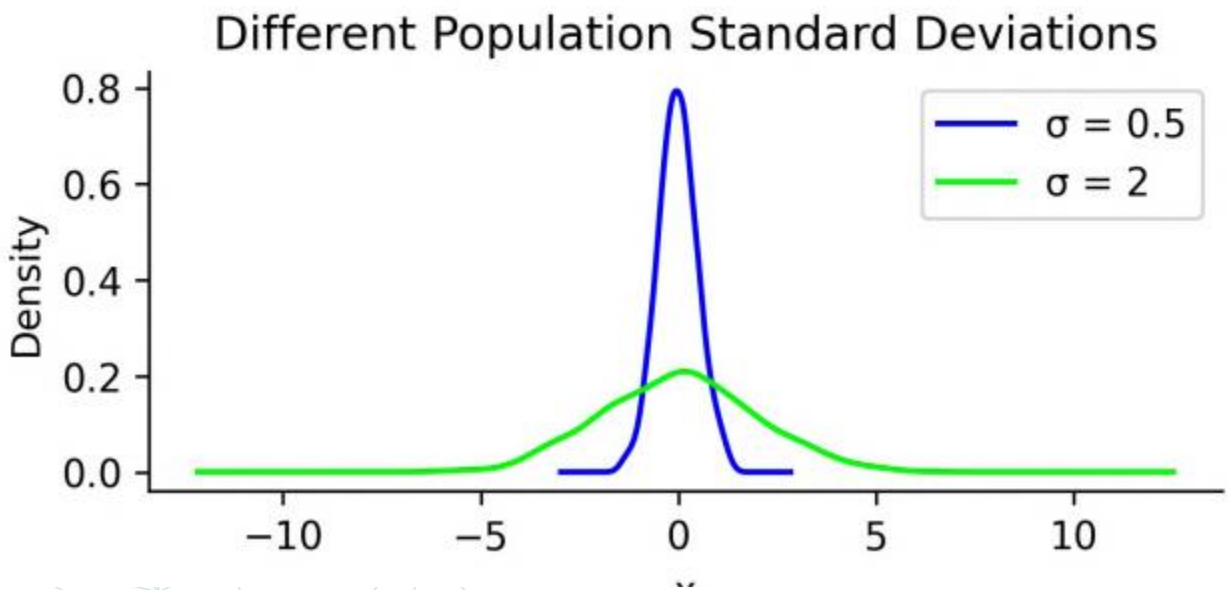
Variance provides a statistic in squared units (e.g., income in $ results in variance in $²), which isn't practical for interpretation. To measure spread in the same units as our data, we use the standard deviation.

Measures of Spread

Standard Deviation

We can use the standard deviation to see how far from the mean data points are on *average*.

A small standard deviation means that values are close to the mean, while a large standard deviation means that values are dispersed more widely.



Measures of Spread

Standard Deviation

The standard deviation is simply the square root of the variance. By performing this operation, we get a statistic in units that we can make sense of again ($ for our income example):

$$s = \sqrt{\frac{\sum_1^n (x_i - \bar{x})^2}{n-1}} = \sqrt{s^2}$$

Measures of Spread

Coefficient of Variation

When we moved from variance to standard deviation, we were looking to get to units that made sense; however, if we then want to compare the level of dispersion of one dataset to another, we will need to have the same units once again.

One way around this is to calculate the **coefficient of variation (CV)**, which is **unitless**. The CV is the ratio of the standard deviation to the mean:

$$CV = \frac{s}{\bar{x}}$$

Measures of Spread

Interquartile Range

So far, other than the range, we have discussed mean-based measures of dispersion; now, we will look at how we can describe the spread with the median as our measure of central tendency.

Percentiles and quartiles are both quantiles—values that divide data into equal groups each containing the same percentage of the total data. Percentiles divide the data into 100 parts, while quartiles do so into four (25%, 50%, 75%, and 100%).

Quantiles divide data into equal sections, making them ideal for measuring spread. The interquartile range (IQR), the distance between the 3rd and 1st quartiles, is a common measure of this spread

$$IQR = Q_3 - Q_1$$

The IQR gives us the spread of data around the median and quantifies how much dispersion we have in the middle 50% of our distribution.

Measures of Spread

Quartile Coefficient of Dispersion

Similar to the coefficient of variation for the mean, the **quartile coefficient of dispersion** is used with the median. It's unitless for comparing datasets and is calculated by dividing the **semi-quartile range** (half the IQR) by the **midhinge** (midpoint between the first and third quartiles):

$$QCD = \frac{\frac{Q_3 - Q_1}{2}}{\frac{Q_1 + Q_3}{2}} = \frac{Q_3 - Q_1}{Q_3 + Q_1}$$

Summarizing Data

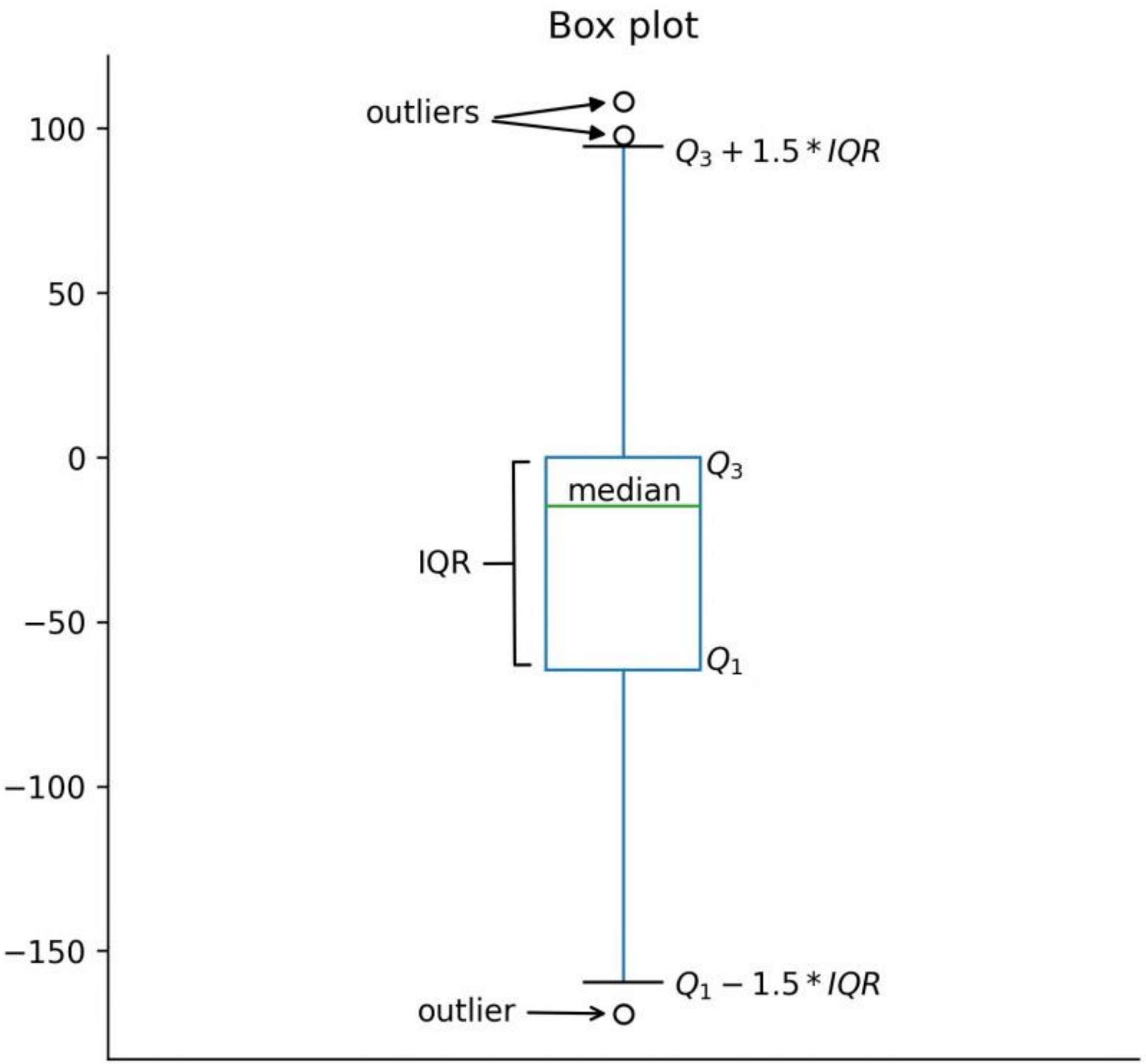
Summarizing data by its center and dispersion often starts with the **5-number summary**, which includes five key statistics and helps visualize the distribution before using other metrics.

	Quartile	Statistic	Percentile
1.	Q_0	minimum	0^{th}
2.	Q_1	N/A	25^{th}
3.	Q_2	median	50^{th}
4.	Q_3	N/A	75^{th}
5.	Q_4	maximum	100^{th}

Summarizing Data

A box plot is a visual representation of the 5-number summary. The median is denoted by a thick line in the box. The top of the box is Q3 and the bottom of the box is Q1.

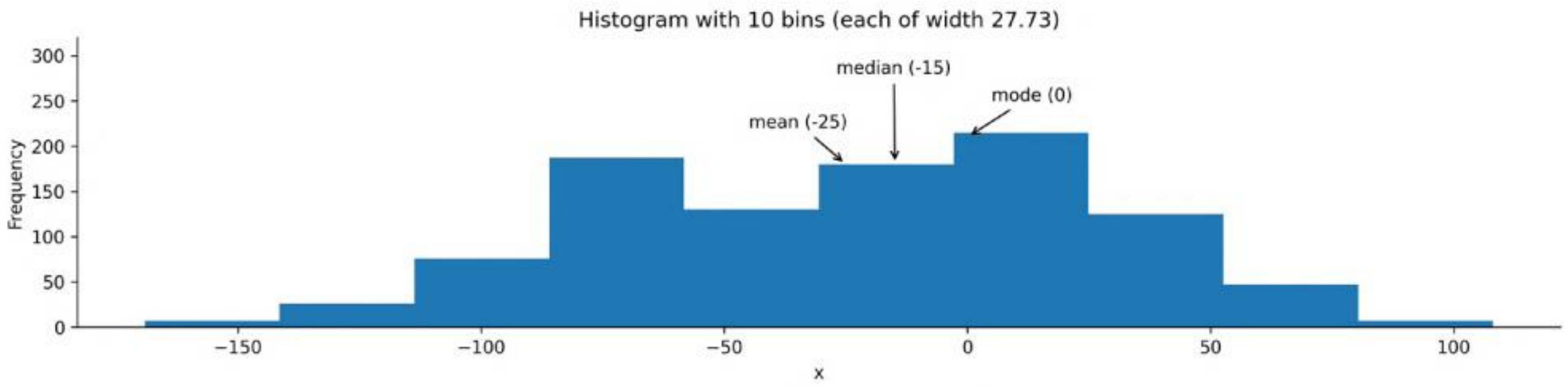
While the box plot is a great tool for getting an initial understanding of the distribution, we don't get to see how things are distributed inside each of the quartiles.



Summarizing Data

Histograms for Discrete Variables

To make a histogram, a certain number of equal-width bins are created, and then bars with heights for the number of values we have in each bin are added.

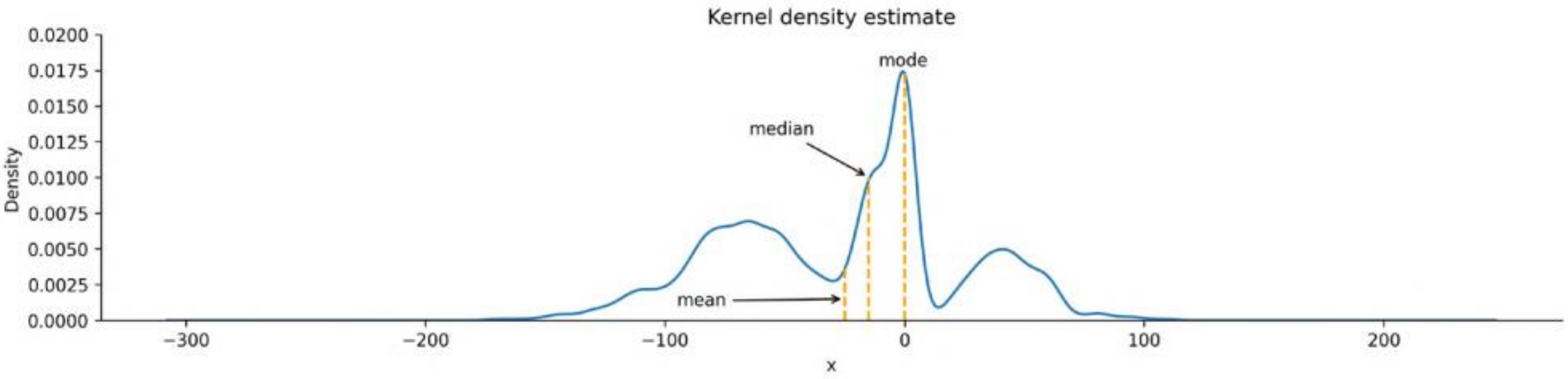


Summarizing Data

Kernel Density Estimates (KDEs) for Continuous Variables

KDEs are similar to histograms, except rather than creating bins for the data, they draw a smoothed curve, which is an estimate of the distribution's probability density function (PDF).

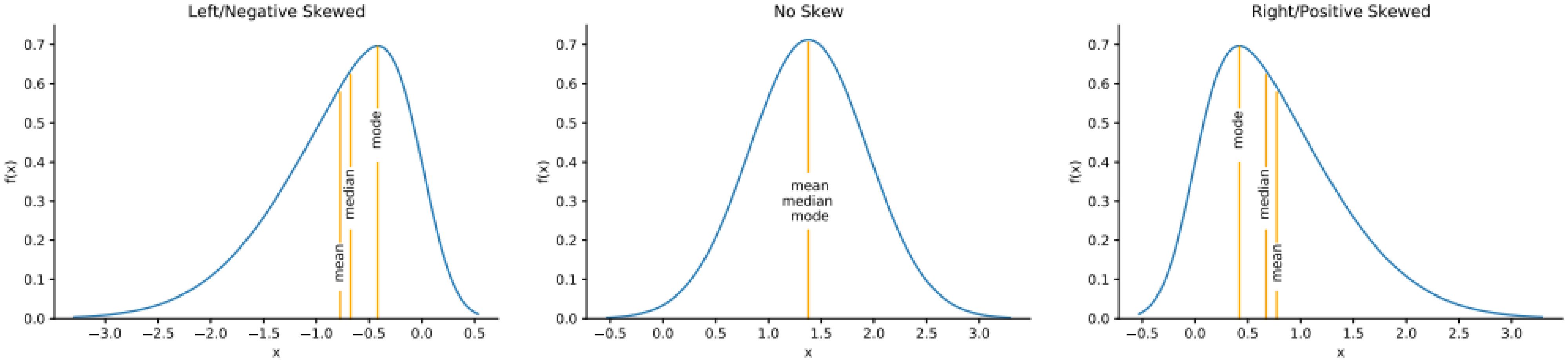
The PDF is for continuous variables and tells us how probability is distributed over the values. Higher values for the PDF indicate higher likelihoods:



Summarizing Data

Skewed Distribution

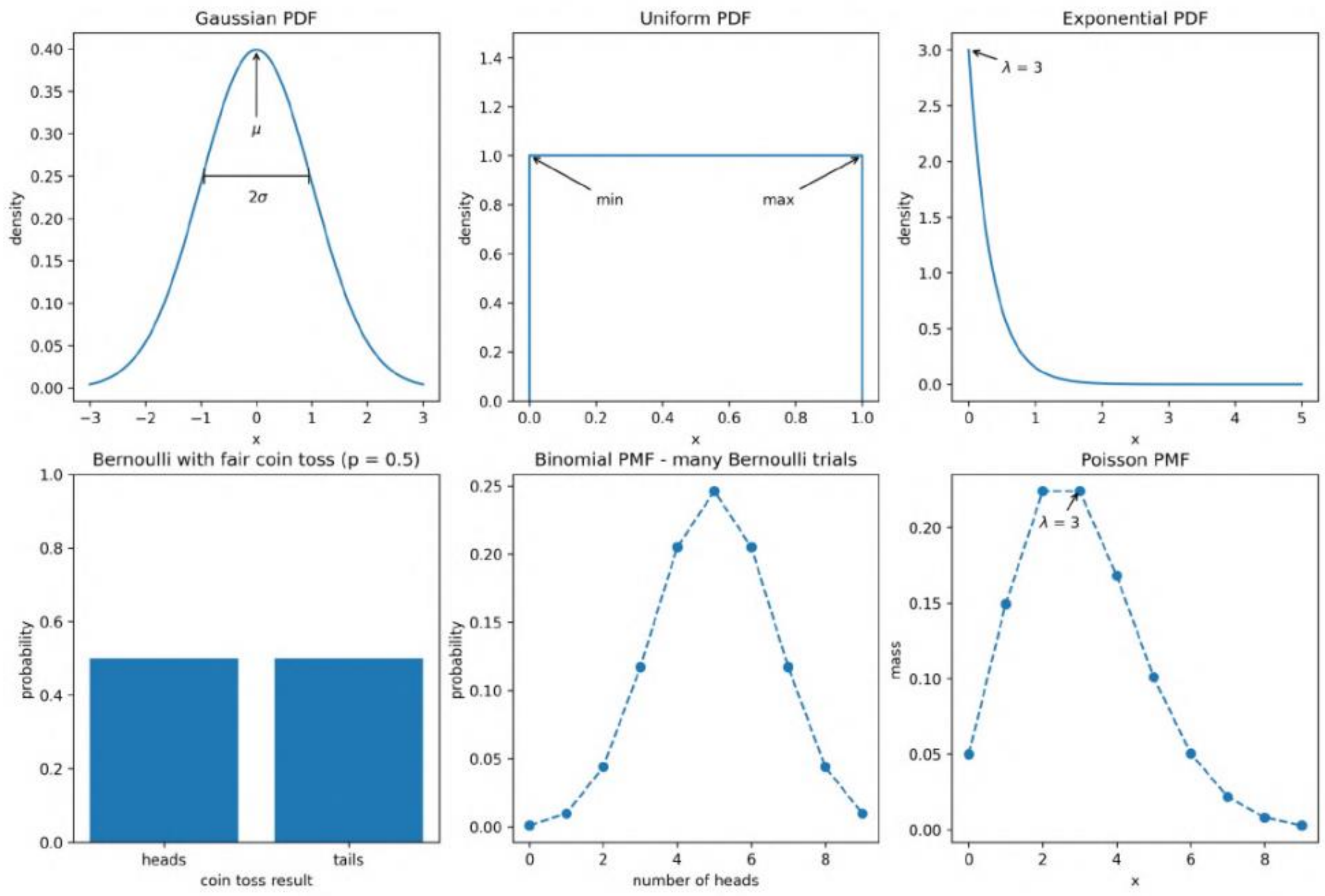
In skewed distributions, the mean can be pulled by long tails. A left (negative) skew has a long left tail, and a right (positive) skew has a long right tail.



Summarizing Data

Common Distribution

Visualizing some commonly used distributions:



Scaling Data

To compare variables from different distributions, we would have to scale the data, which we could do with the range by using min-max scaling.

$$x_{scaled} = \frac{x - \min(X)}{range(X)}$$

This isn't the only way to scale data; we can also use the mean and standard deviation (**standardize**).

$$z_i = \frac{x_i - \bar{x}}{s}$$

There are, of course, additional ways to scale our data, and the one we end up choosing will be dependent on our data and what we are trying to do with it.

Quantifying Relationships Between Variables

Univariate statistics focus on a single variable. Multivariate statistics, however, quantify relationships between variables and help predict future behavior

Covariance

The covariance is a statistic for quantifying the relationship between variables by showing how one variable changes with respect to another (also referred to as their joint variance):

$$cov(X,Y) = E[(X - E[X])(Y - E[Y])]$$

Covariance indicates whether variables are positively or negatively correlated, but its magnitude is hard to interpret. To quantify the strength of the relationship, we use correlation.

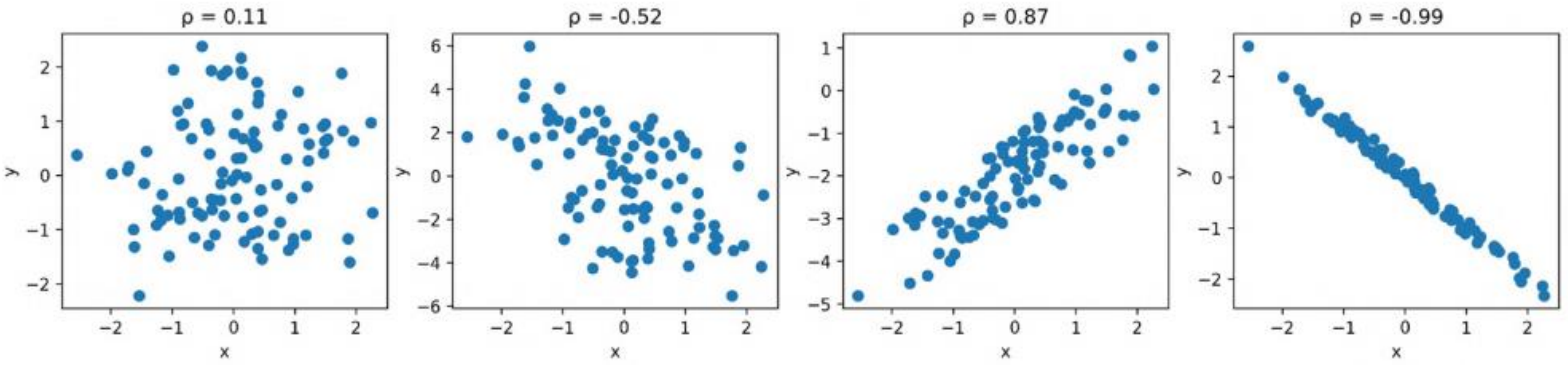
Correlation

Correlation tells us how variables change together both in direction (same or opposite) and magnitude (strength of the relationship).

$$\rho_{X,Y} = \frac{cov(X,Y)}{s_X s_Y}$$

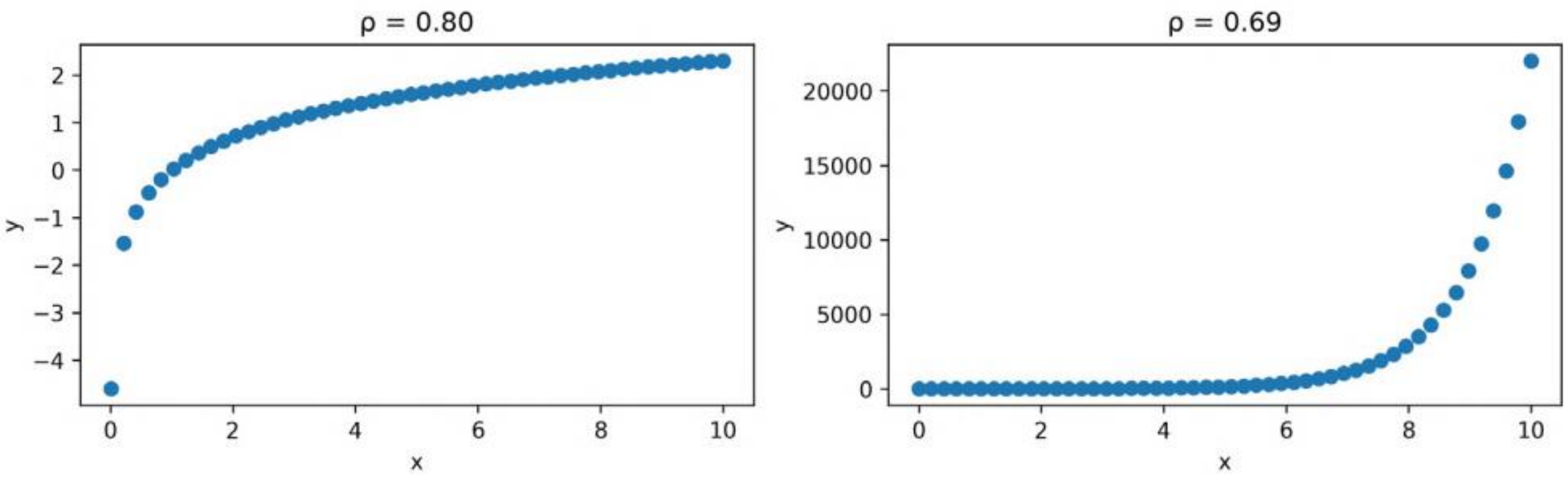
Quantifying Relationships Between Variables

- Normalizing covariance gives a correlation statistic between -1 and 1, indicating direction (sign) and strength (magnitude).
- A correlation of 1 means a perfect positive relationship, -1 means a perfect negative relationship, and values near 0 indicate no correlation.
- Values near 1 or -1 suggest strong correlation, while those near 0.5 suggest weak correlation.



Quantifying Relationships Between Variables

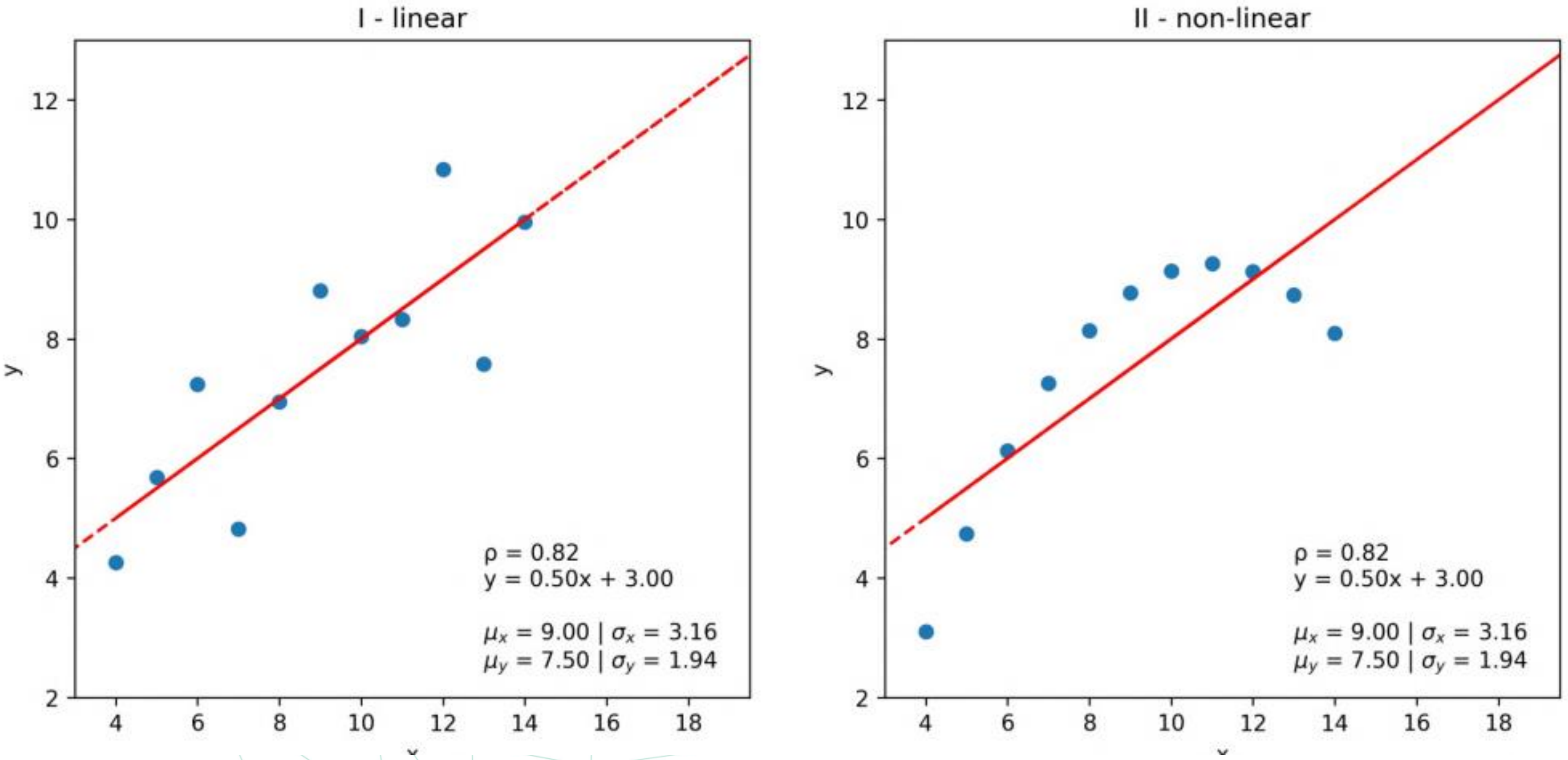
Both of the following plots depict data with strong positive correlations, but it's pretty obvious when looking at the scatter plots that these are not linear.



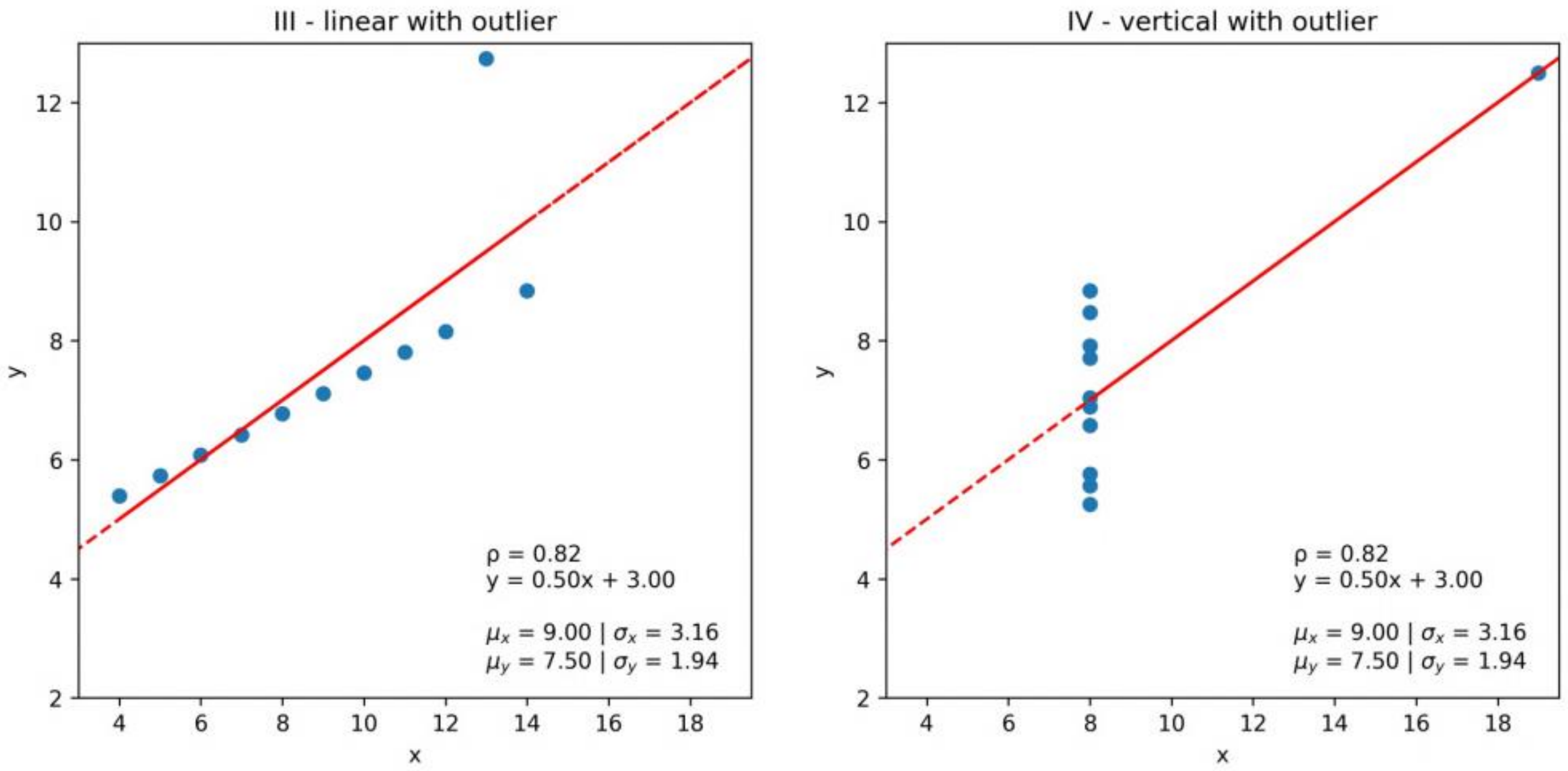
It's very important to remember that while we may find a correlation between X and Y, it doesn't mean that X causes Y or that Y causes X. There could be some Z that actually causes both.

Pitfalls of Summary Statistics

Anscombe's quartet demonstrates the need for plotting data. Despite identical summary statistics and correlation coefficients, the four datasets are clearly different when visualized.



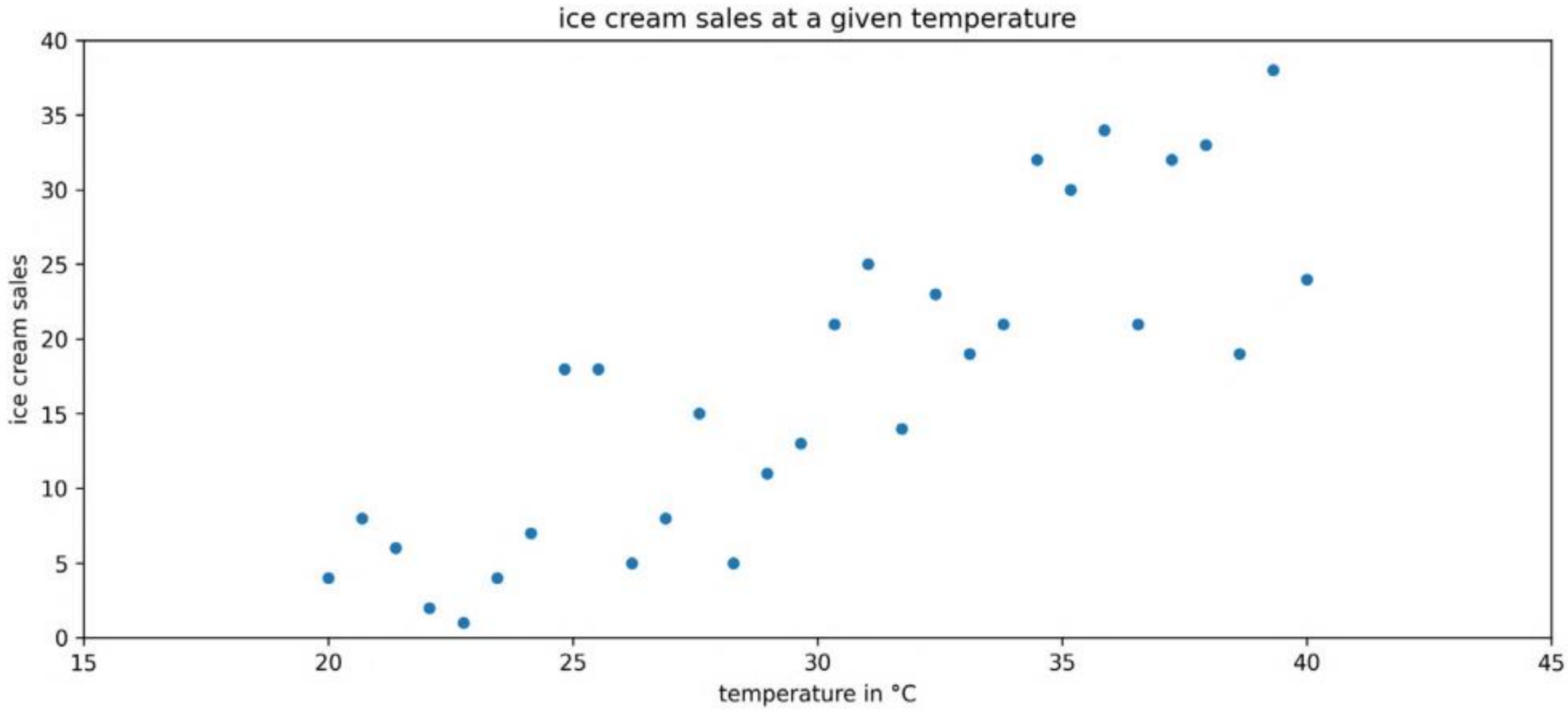
Pitfalls of Summary Statistics



Summary statistics are very helpful when we're getting to know the data but be wary of relying exclusively on them. Remember, statistics can be misleading; be sure to also plot the data before drawing any conclusions or proceeding with the analysis.

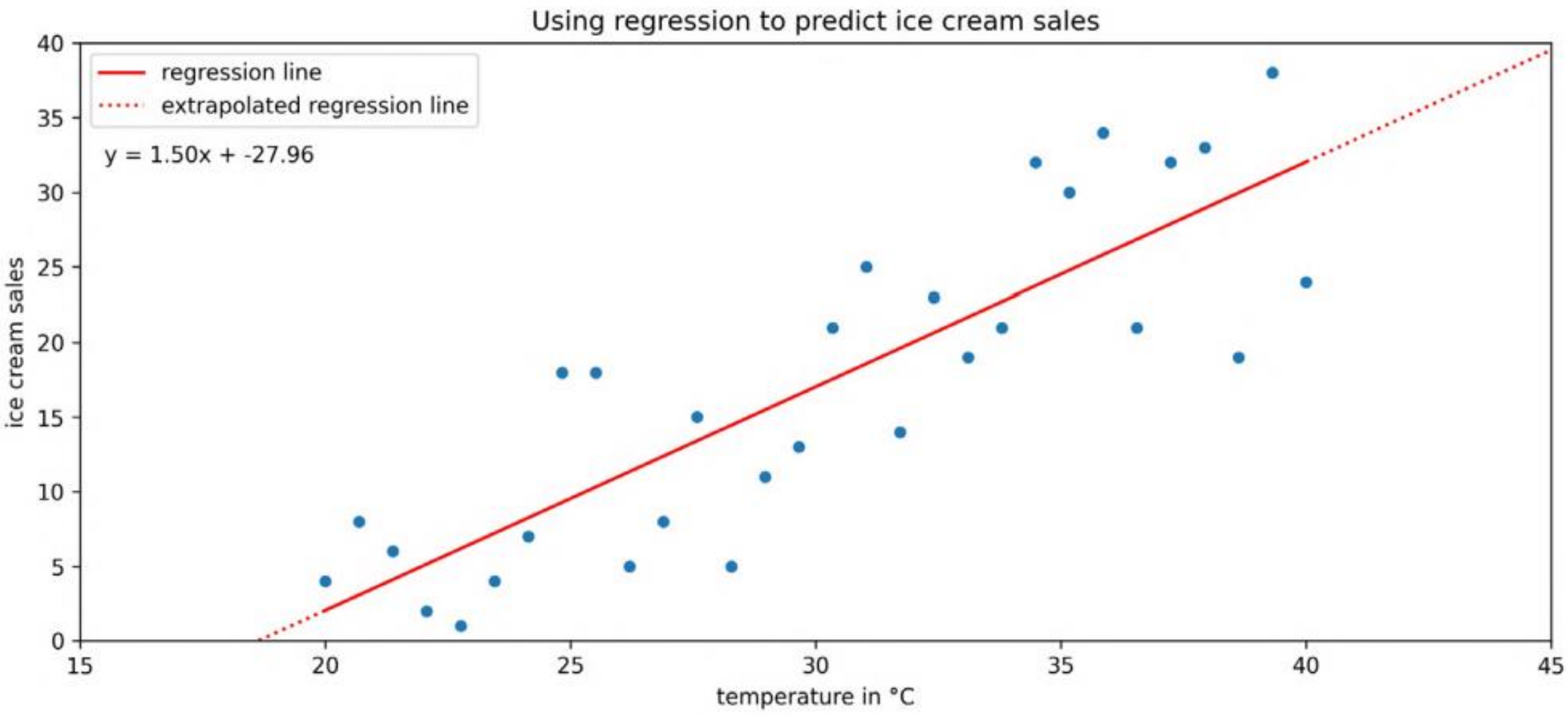
Prediction and Forecasting

Say our favorite ice cream shop has asked us to help predict how many ice creams they can expect to sell on a given day.



Prediction and Forecasting

An upward trend in the scatter plot shows more ice creams sold at higher temperatures. To make predictions, we use regression to model the relationship between temperature and ice cream sales.



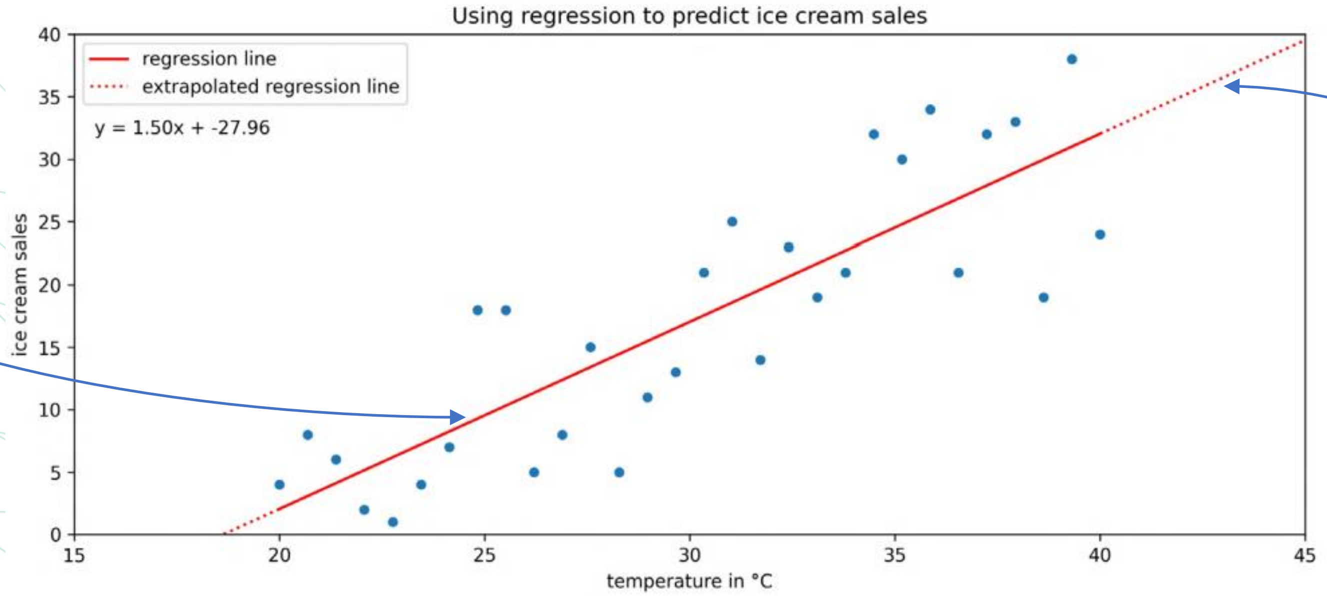
Prediction and Forecasting

The regression line in the previous scatter plot yields the following equation for the relationship:

$$ice\ cream\ sales = 1.50 \times temperature - 27.96$$

Suppose that today the temperature is 35°C—we would plug that in for temperature in the equation. The result predicts that the ice cream shop will sell 24.54 ice creams.

Interpolation



Extrapolation

Prediction and Forecasting

In time series, **forecasting** predicts future values based on past ones. We use time series decomposition to split data into components, which are combined additively or multiplicatively for modeling.

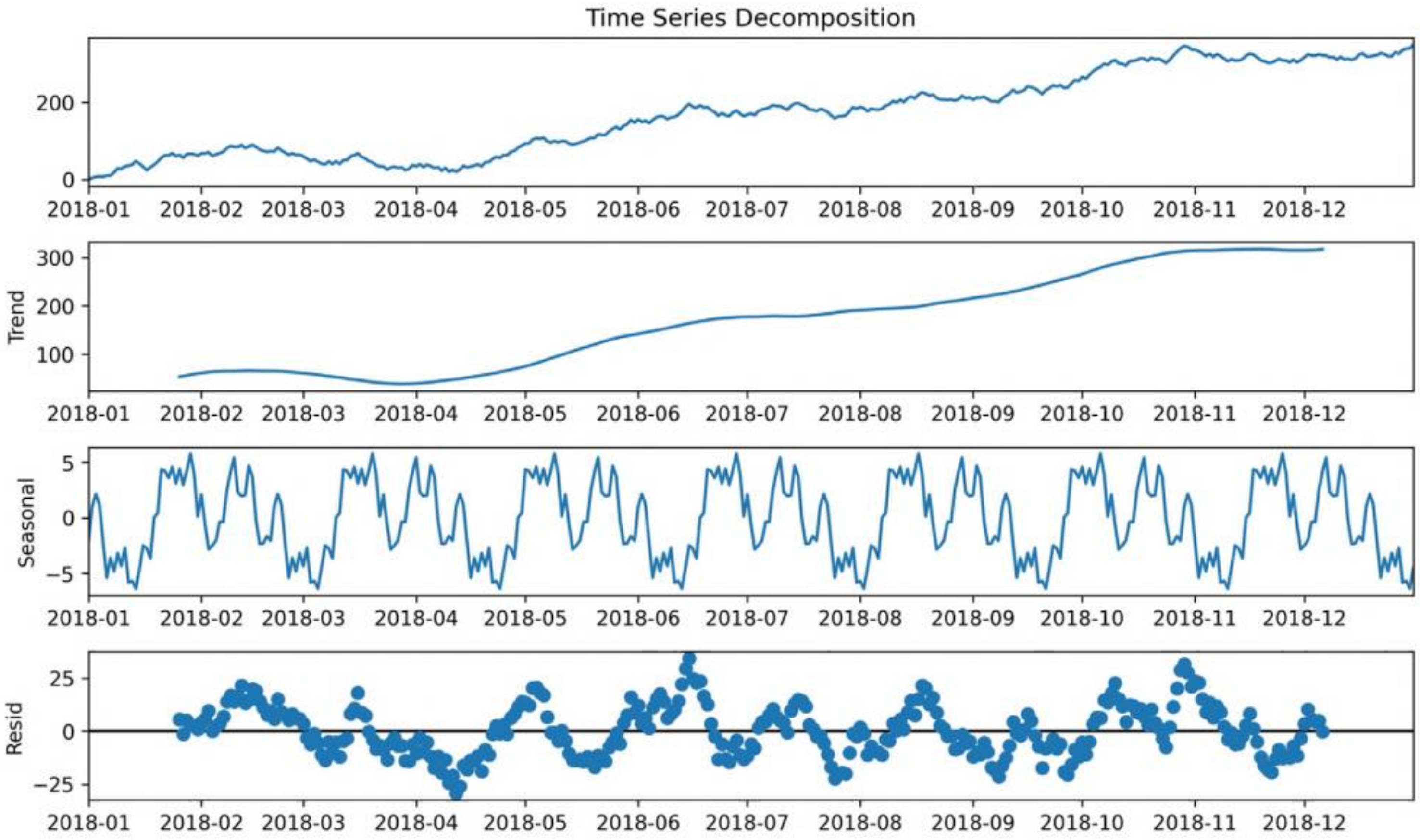
The **trend** component shows **long-term** behavior without seasonal or cyclical effects, allowing broad statements about long-term patterns, such as Earth's population increasing or stock value stagnating.

The **seasonality** component explains the systematic and calendar-related movements of a time series. For example, the number of ice cream trucks on the streets of New York City is high in the summer and drops to nothing in the winter.

The **cyclical** component captures irregular or unexplained events, like a hurricane temporarily reducing the number of ice cream trucks due to unsafe conditions.

Prediction and Forecasting

We can decompose time series into **trend**, **seasonality**, and **residuals** (noise). The **cyclical** component is represented in the residuals (random, unpredictable data) after removing trend and seasonality.



EDA | Exploratory Data Analysis in Python

- Exploratory data analysis, or EDA, is a crucial step in the data analysis process that involves studying, exploring, and visualizing information to derive important insights. To find patterns, trends, and relationships in the data, it makes use of statistical tools and visualizations. This helps to formulate hypotheses and direct additional investigations.

- Python provides strong EDA tools with its diverse library ecosystem, which includes Seaborn, Matplotlib, and Pandas. An essential phase in the data science pipeline, this procedure improves data comprehension and provides information for further modeling decisions.

EDA | Exploratory Data Analysis in Python

- Exploratory Data Analysis(EDA) is the main step in the process of various data analysis. It helps data to visualize the patterns, characteristics, and relationships between variables. Python provides various libraries used for EDA such as NumPy, Pandas, Matplotlib, Seaborn, and Plotly.

What is Exploratory Data Analysis (EDA)?

- EDA is a phenomenon under data analysis used for gaining a better understanding of data aspects like:

  - main features of data
  - variables and relationships that hold between them
  - Identifying which variables are important for our problem

What is Exploratory Data Analysis (EDA)?

We shall look at various exploratory data analysis methods like:

- Reading dataset
- Analyzing the data
- Checking for the duplicates
- Missing Values Calculation
- Exploratory Data Analysis
    - Univariate Analysis
    - Bivariate Analysis
    - Multivariate Analysis

What is Preprocessing and Data Engineering?

- When referring to data preparation and cleaning, preprocessing is done before raw data is entered into an analytical tool or machine learning model. Missing value handling, feature scaling, categorical variable encoding, and outlier removal are all part of it. To improve the performance and interpretability of the model, it is important to make sure the data is in the right format. Data-driven jobs are more successful overall when preprocessing is used to reduce noise, standardize data, and optimize it for effective analysis.
- The practical application of ideas, techniques, and technology for gathering, storing, analyzing, and organizing massive amounts of data is known as data engineering. It includes building reliable data architectures, constructing data pipelines, and putting in place mechanisms that make information flow easier. Data engineers ensure data quality, dependability, and accessibility while building the infrastructure needed to support data-driven applications. Data engineering is a fundamental component of the larger data science and analytics ecosystem because it helps firms extract meaningful insights from their data.

وزارة الاتصـــــــــالات وتكنولوجيا المعلومات

رواد مصر الرقمية

Step 1: Importing Required Libraries

Understanding and experimenting with our data using libraries is the first step in utilizing Python for machine learning. The dataset can be accessed via this link.

Import all of the libraries needed for our investigation, including those for data loading, statistical analysis, visualizations, univariate and bivariate analysis, etc.

```
# importting Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings as wr
wr.filterwarnings('ignore')
```

Step 2: Reading Dataset

```
# loading and reading dataset
df = pd.read_csv("winequality-red.csv")
print(df.head())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076
   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56
   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

Step 3:Analyzing the Data

Gaining general knowledge about the data—including its values, kinds, number of rows and columns, and missing values—is the primary objective of data understanding.

- shape: shape will show how many features (columns) and observations (rows) there are in the dataset.

```
# shape of the data
df.shape
```

```
(1599, 12)
```

Step 3: Analyzing the Data

- info() facilitates comprehension of the data type and related information, such as the quantity of records in each column, whether the data is null or not, the type of data, and the dataset's memory use.

```
#data information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Step 3: Analyzing the Data

- The DataFrame "df" is statistically summarized by the code df.describe(), which gives the count, mean, standard deviation, minimum, and quartiles for each numerical column. The dataset's central tendencies and spread are briefly summarized.

```
# describing the data
df.describe()
```

```
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1599.000000       1599.000000  1599.000000     1599.000000
mean        8.319637          0.527821     0.270976        2.538806
std         1.741096          0.179060     0.194801        1.409928
min         4.600000          0.120000     0.000000        0.900000
25%         7.100000          0.390000     0.090000        1.900000
50%         7.900000          0.520000     0.260000        2.200000
75%         9.200000          0.640000     0.420000        2.600000
max        15.900000          1.580000     1.000000       15.500000
         chlorides  free sulfur dioxide  total sulfur dioxide      density  \
count  1599.000000          1599.000000           1599.000000  1599.000000
mean      0.087467            15.874922             46.467792     0.996747
std       0.047065            10.460157             32.895324     0.001887
min       0.012000             1.000000              6.000000     0.990070
25%       0.070000             7.000000             22.000000     0.995600
50%       0.079000            14.000000             38.000000     0.996750
75%       0.090000            21.000000             62.000000     0.997835
max       0.611000            72.000000            289.000000     1.003690
                pH    sulphates      alcohol
count  1599.000000  1599.000000  1599.000000
mean      3.311113     0.658149    10.422983
std       0.154386     0.169507     1.065668
min       2.740000     0.330000     8.400000
25%       3.210000     0.550000     9.500000
50%       3.310000     0.620000    10.200000
75%       3.400000     0.730000    11.100000
max       4.010000     2.000000    14.900000
```

Step 3: Analyzing the Data

- The code df.columns.tolist() converts the column names of the DataFrame 'df' into a Python list, providing a convenient way to access and manipulate column names.

```
#column to list
df.columns.tolist()
```

```
['fixed acidity',
 'volatile acidity',
 'citric acid',
 'residual sugar',
 'chlorides',
 'free sulfur dioxide',
 'total sulfur dioxide',
 'density',
 'pH',
 'sulphates',
 'alcohol',
 'quality']
```

Step 3: Analyzing the Data

- The code df.isnull().sum() checks for missing values in each column of the DataFrame 'df' and returns the sum of null values for each column.

```
# check for missing values:
df.isnull().sum()
```

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Step 3: Analyzing the Data

- The function df.nunique() determines how many unique values there are in each column of the DataFrame "df," offering information about the variety of data that makes up each feature.

```
#checking duplicate values
df.nunique()
```

```
fixed acidity            96
volatile acidity        143
citric acid              80
residual sugar           91
chlorides               153
free sulfur dioxide      60
total sulfur dioxide    144
density                 436
pH                       89
sulphates                96
alcohol                  65
quality                   6
dtype: int64
```

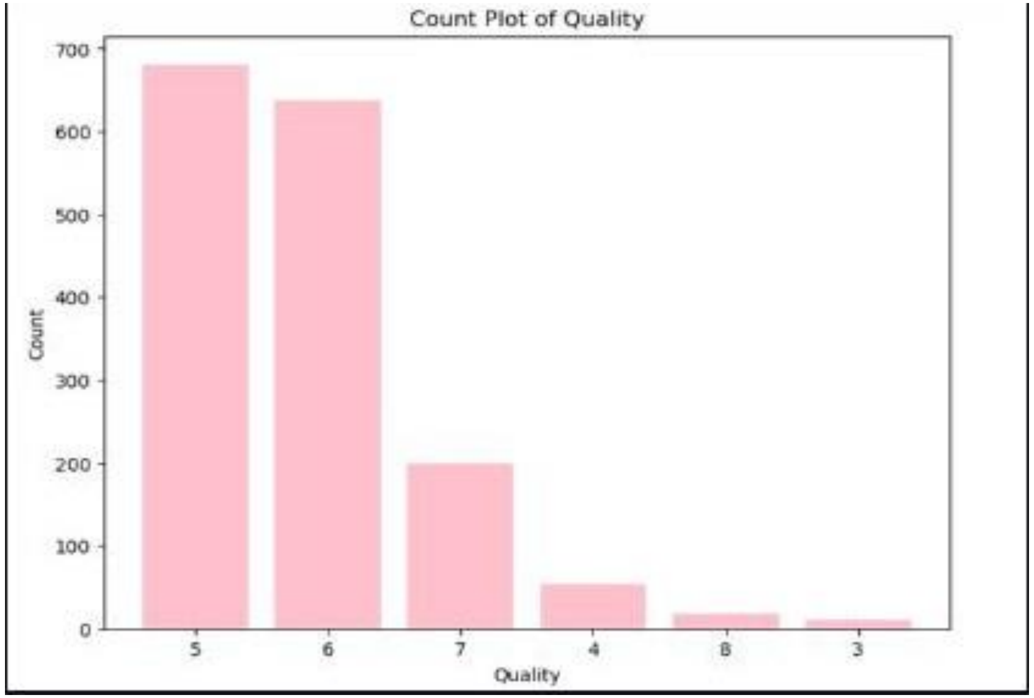
Step 4: Univariate Analysis

- In Univariate analysis, plotting the right charts can help us better understand the data, which is why data visualization is so important. Matplotlib and Seaborn libraries are used in this post to visualize our data.
- For both numerical and categorical data, univariate analysis is an option.
- In this example, we are going to plot different types of plots like swarmplots, violinplots, and countplots for univariate analysis.

Step 4: Univariate Analysis

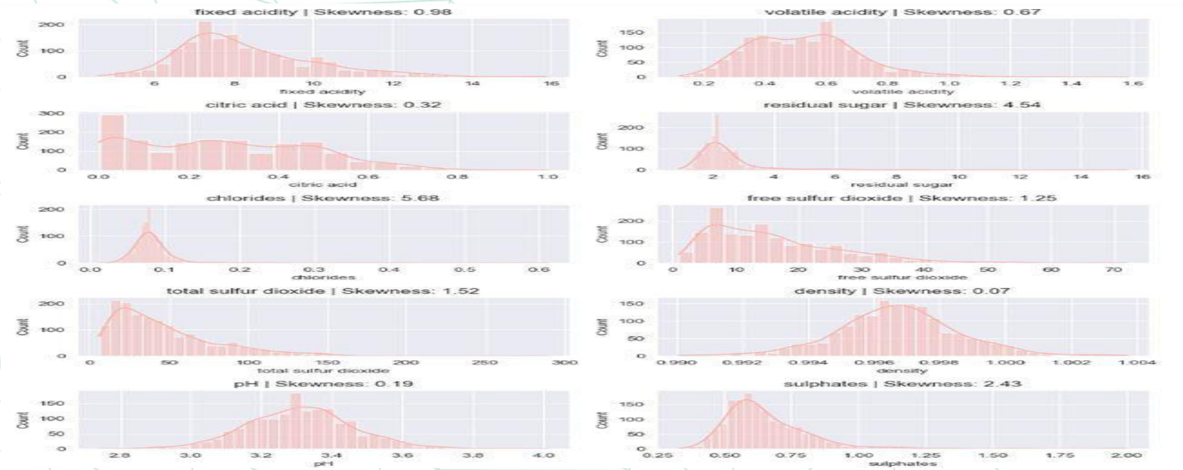
❑ Here , this count plot graph shows the count of the wine with its quality rate.

```
# Assuming 'df' is your DataFrame
quality_counts = df['quality'].value_counts()

# Using Matplotlib to create a count plot
plt.figure(figsize=(8, 6))
plt.bar(quality_counts.index, quality_counts, color='darpink')
plt.title('Count Plot of Quality')
plt.xlabel('Quality')
plt.ylabel('Count')
plt.show()
```



Step 4: Univariate Analysis

- Here, in the kernel density plot is about the skewness of the of the corresponding feature. The features in this dataset that have skewness are exactly 0 depicts the symmetrical distribution and the plots with skewness 1 or above 1 is positively or right skewd distribution.

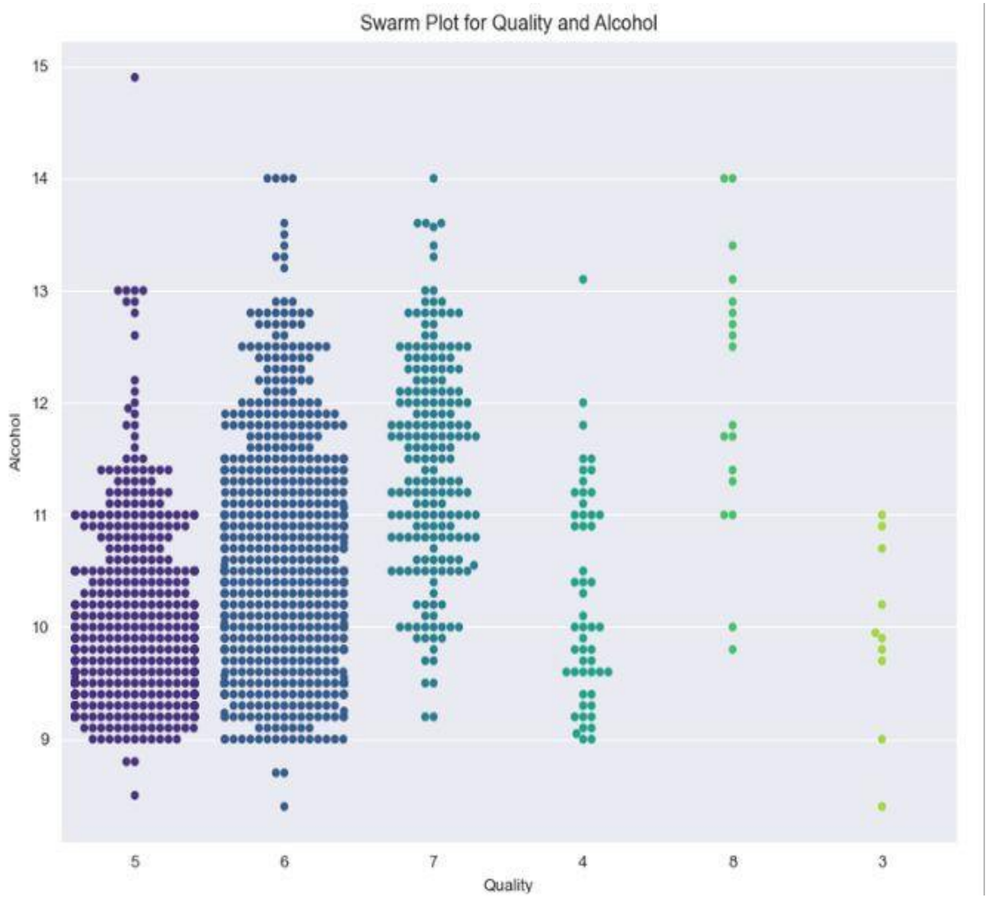



```python
# Set Seaborn style
sns.set_style("darkgrid")

# Identify numerical columns
numerical_columns = df.select_dtypes(include=["int64", "float64"]).columns

# Plot distribution of each numerical feature
plt.figure(figsize=(14, len(numerical_columns) * 3))
for idx, feature in enumerate(numerical_columns, 1):
    plt.subplot(len(numerical_columns), 2, idx)
    sns.histplot(df[feature], kde=True)
    plt.title(f"{feature} | Skewness: {round(df[feature].skew(), 2)}")

# Adjust layout and show plots
plt.tight_layout()
plt.show()
```

Step 4: Univariate Analysis

- This graph shows the swarm plot for 'Quality' and 'Alcohol' column. This plot depicts that the higher point density in specific regions shows the concentration indicating where the majority of data points cluster. The points isolated and are far away from the clusters shows the outliers.

```
# Assuming 'df' is your DataFrame
plt.figure(figsize=(10, 8))

# Using Seaborn to create a swarm plot
sns.swarmplot(x="quality", y="alcohol", data=df, palette='viridis')

plt.title('Swarm Plot for Quality and Alcohol')
plt.xlabel('Quality')
plt.ylabel('Alcohol')
plt.show()
```



Step 5: Bivariate Analysis

- When doing a bivariate analysis, two variables are examined simultaneously in order to look for patterns, dependencies, or interactions between them. Understanding how changes in one variable may correspond to changes in another requires the use of this statistical method.

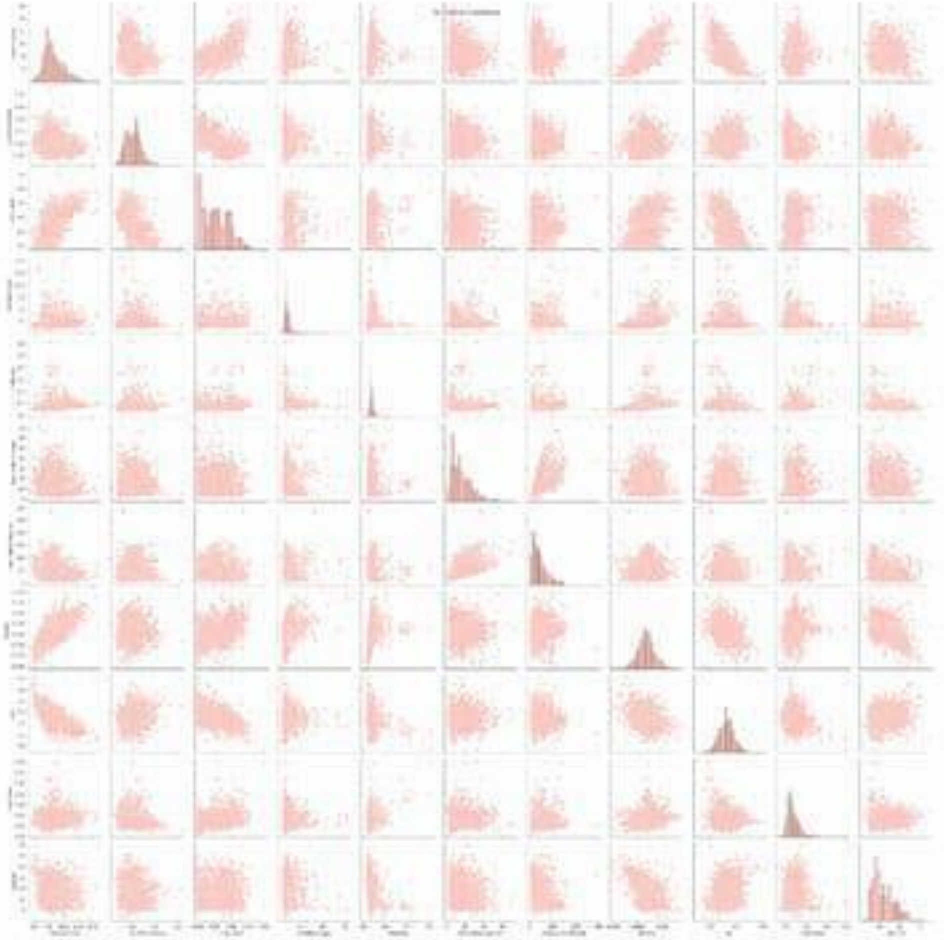
- Bivariate analysis allows for a thorough comprehension of the interdependence between two variables within a dataset by revealing information on the type and intensity of associations.

- Let's plot a pair plot for the data.

Step 5: Bivariate Analysis

- Pair Plot
  - If the plot is diagonal , histograms of kernel density plots , is shows the distribution of the individual variables.
  - If the scatter plot is in the lower triangle, it displays the relationship between the pairs of the variables.
  - If the scatter plots above and below the diagonal are mirror images, indicating symmetry.
  - If the histogram plots are more centered, it represents the locations of peaks.
  - Skewness is depicted by observing whether the histogram is symmetrical or skewed to the left or right.

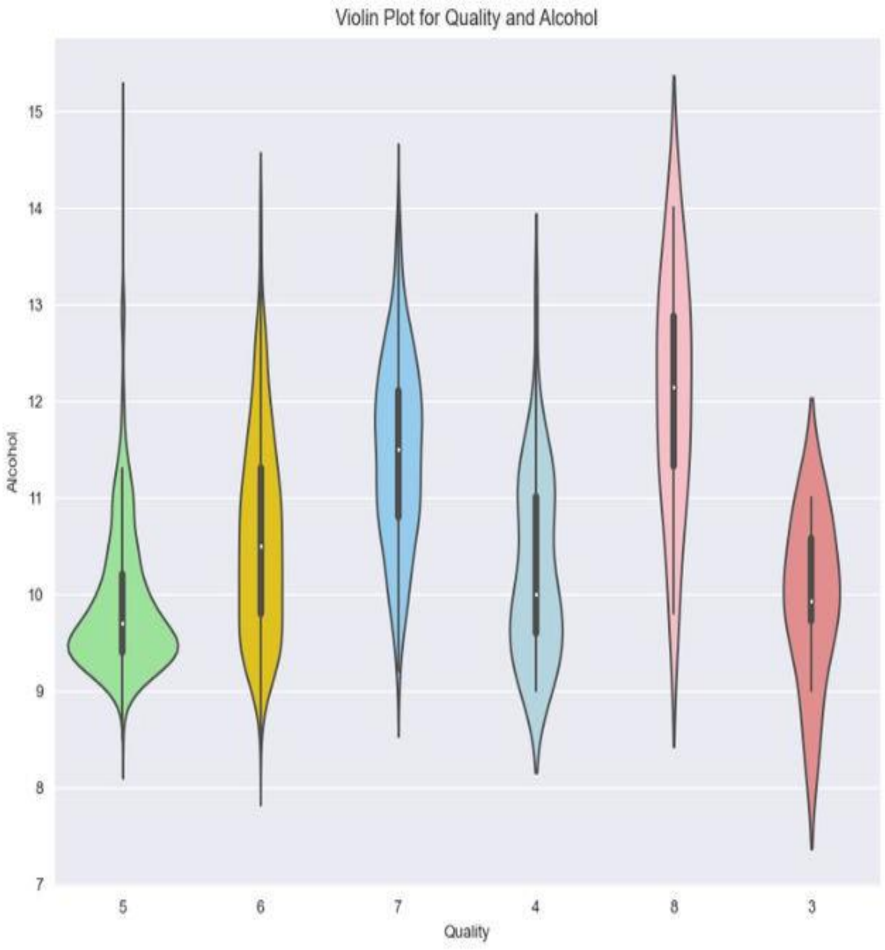
```
# Set the color palette
sns.set_palette("Pastel1")

# Assuming 'df' is your DataFrame
plt.figure(figsize=(10, 6))

# Using Seaborn to create a pair plot with the specified color palette
sns.pairplot(df)

plt.suptitle('Pair Plot for DataFrame')
plt.show()
```



Step 5: Bivariate Analysis

- Violin Plot
  - If the width is wider, it indicates higher density, suggesting more data points.
  - Symmetrical plot indicates a balanced distribution.
  - Peak or bulge in the violin plot represents most common value in distribution.
  - Longer tails indicate great variability.
  - Median line is the middle line inside the violin plot. It helps in understanding central tendencies.

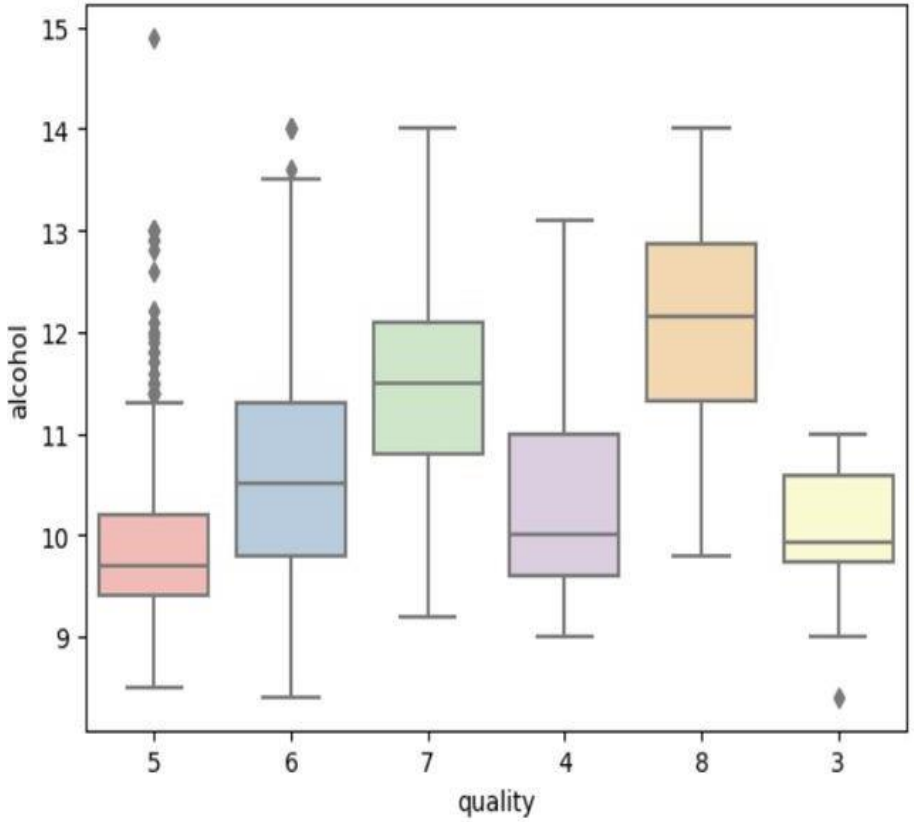
```
# Assuming 'df' is your DataFrame
df['quality'] = df['quality'].astype(str)  # Convert 'quality' to categorical

plt.figure(figsize=(10, 8))

# Using Seaborn to create a violin plot
sns.violinplot(x="quality", y="alcohol", data=df, palette={
           '3': 'lightcoral', '4': 'lightblue', '5': 'lightgreen', '6': 'gold', '7': 'lightskyblue', '8': 'lightpink'}, alpha=0.7)

plt.title('Violin Plot for Quality and Alcohol')
plt.xlabel('Quality')
plt.ylabel('Alcohol')
plt.show()
```



Step 5: Bivariate Analysis

- Box Plot
  - Box represents the IQR. Longer the box, greater the variability.
  - The median line in the box indicates central tendency.
  - Whiskers extend from box to the smallest and largest values within a specified range.
  - Individual points beyond the whiskers represents outliers.
  - A compact box indicates low variability while a stretched box indicates higher variability.

```
#plotting box plot between alcohol and quality
sns.boxplot(x='quality', y='alcohol', data=df)
```



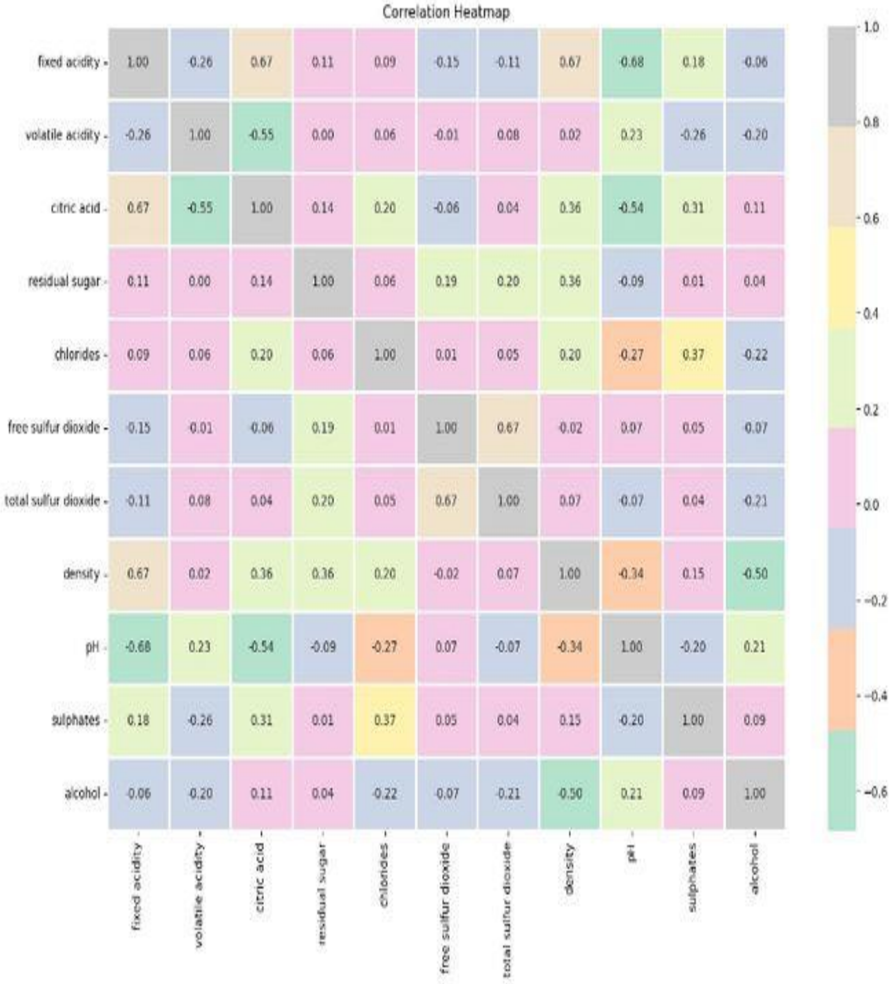
Step 6: Multivariate Analysis

- Interactions between three or more variables in a dataset are simultaneously analyzed and interpreted in multivariate analysis.
- In order to provide a comprehensive understanding of the collective behavior of several variables, it seeks to reveal intricate patterns, relationships, and interactions between them.
- Multivariate analysis examines correlations and dependencies between numerous variables by using sophisticated statistical techniques such factor analysis, principal component analysis, and multivariate regression.
- Multivariate analysis, which is widely applied in domains such as biology, economics, and marketing, enables thorough insights and helps decision-makers make well-informed judgments based on complex relationships found in multidimensional datasets.
- Here, we are going to show the multivariate analysis using a correlation matrix plot.

Step 6: Multivariate Analysis

- ❑ Correlation Matrix
  - ❑ Values close to +1 indicates strong positive correlation, -1 indicates a strong negative correlation and 0 indicates suggests no linear correlation.
  - ❑ Darker colors signify strong correlation, while light colors represents weaker correlations.
  - ❑ Positive correlation variable move in same directions. As one increases, the other also increases.
  - ❑ Negative correlation variable move in opposite directions. An increase in one variable is associated with a decrease in the other.

```python
# Assuming 'df' is your DataFrame
plt.figure(figsize=(15, 10))

# Using Seaborn to create a heatmap
sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='Pastel2', linewidths=2)

plt.title('Correlation Heatmap')
plt.show()
```



Conclusion

In summary, the Python-based exploratory data analysis (EDA) of the wine dataset has yielded important new information about the properties of the wine samples. We investigated correlations between variables, identified outliers, and obtained a knowledge of the distribution of important features using statistical summaries and visualizations. The quantitative and qualitative features of the dataset were analyzed in detail through the use of various plots, including pair, box, and histogram plots. Finding patterns, trends, and possible topics for more research was made easier by this EDA method. Furthermore, the analysis demonstrated the ability to visualize and analyze complicated datasets using Python tools such as Matplotlib, Seaborn, and Pandas. The results provide a thorough grasp of the wine dataset and lay the groundwork for more in-depth studies and modeling.

Q&A

Questions and answers

Thanks