CoGrammar

Welcome to this session

Skills Bootcamp:

Exploratory Data Analysis (Theory)

The session will start shortly...

Questions? Drop them in the chat.
We'll have dedicated moderators
answering questions.

Skills Bootcamp Data Science Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. We will be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Skills Bootcamp Data Science Housekeeping

- For all **non-academic questions**, please submit a query: ***www.hyperiondev.com/support***

- **Report a safeguarding incident: *www.hyperiondev.com/safeguardreporting***

- We would love your feedback on lectures: ***Feedback on Lectures.***

- Find all the lecture **content** in your **Lecture Backpack** on GitHub.

- If you are hearing impaired, kindly use your computer's function through Google chrome to enable captions.

CoGrammar

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Skills Bootcamp Progression Overview

✅ Criterion 1 - Initial Requirements

Specific achievements within the first two weeks of the program.

To meet this criterion, students need to, by no later than **01 December 2024 (C11)** or **22 December 2024 (C12)**:

- **Guided Learning Hours** (GLH): Attend a minimum of 7-8 GLH per week (lectures, workshops, or mentor calls) for a total minimum of **15 GLH**.
- **Task Completion:** Successfully complete the **first 4 of the assigned tasks**.

✅ Criterion 2 - Mid-Course Progress

Progress through the successful completion of tasks within the first half of the program.

To meet this criterion, students should, by no later than **12 January 2025 (C11)** or **02 February 2025 (C12)**:

- **Guided Learning Hours** (GLH): Complete at least **60 GLH**.
- **Task Completion :** Successfully complete the **first 13 of the assigned tasks**.

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3 – End-Course Progress

Showcasing students' progress nearing the completion of the course.

To meet this criterion, students should:

- **Guided Learning Hours** (GLH)**:** Complete the **total minimum required GLH,** by the **support end date.**
- **Task Completion : Complete all mandatory tasks**, including any necessary resubmissions, by the end of the bootcamp, **09 March 2025 (C11)** or **30 March 2025 (C12).**

✅ Criterion 4 - Employability

Demonstrating progress to find employment.

To meet this criterion, students should:

- **Record an Interview Invite:** Students are required to record proof of invitation to an interview by **30 March 2025 (C11)** or **04 May 2025 (C12)**.
    - **South Holland Students** are required to proof and interview by **17 March 2025.**
- **Record a Final Job Outcome :** Within 12 weeks post-graduation, students are required to record a job outcome.

CoGrammar

Learning Outcomes

- Understand and apply Exploratory Data Analysis (EDA) techniques to effectively analyse datasets.
- Understand the importance of EDA in data science projects
- Apply EDA techniques to clean, preprocess, and explore data
- Use Python libraries for EDA tasks and data visualisation

Lecture Overview

- ➔ Understanding EDA – Learn what EDA is, why it’s important, and how it fits into the data science workflow.
- ➔ Key EDA Techniques – Explore univariate, bivariate, and multivariate analysis.
- ➔ Applying EDA in Practice – Discuss real-world applications, and how EDA guides data-driven decision-making.

Introduction to Exploratory Data Analysis

Image source: https://www.questionpro.com/blog/exploratory-data-analysis/

CoGrammar

Real-World Application of EDA

Imagine you're an analyst for an online store.

You have sales data, including product categories, prices, and customer locations. The management wants to know:

- Which products are top sellers?
- Are there seasonal sales trends?
- How do customer preferences vary by region?

Using EDA, you can uncover patterns, identify trends, and find outliers to guide marketing and inventory decisions.

Introduction to EDA

- **Definition:** Exploratory Data Analysis (EDA) is the process of investigating and understanding a dataset through visual and statistical techniques.
- **Importance:** EDA is a crucial first step in any data science project as it helps uncover patterns, anomalies, and relationships in the data, guiding further analysis and decision-making.
- **Role in the data science workflow:** EDA is performed after data collection and before model building and evaluation. It helps in understanding the data, identifying data quality issues, and selecting relevant features for modeling.

Simple EDA Framework

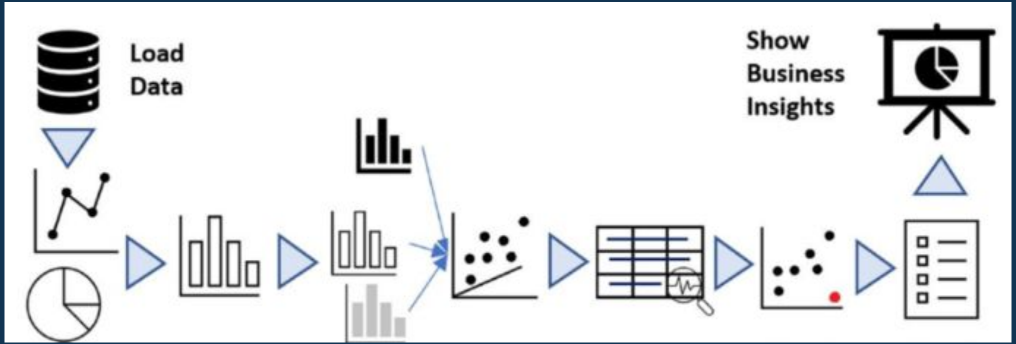


Image source: https://www.appliedaicourse.com/blog/exploratory-data-analysis/

Simple EDA Framework

When performing exploratory data analysis, we typically follow most of these five key steps in some order:

1. Understand the data.
2. Clean and preprocess the data.
3. Explore relationships.
4. Assess feature importance.
5. Iterate and refine.

Simple EDA Framework

Understand the Data:

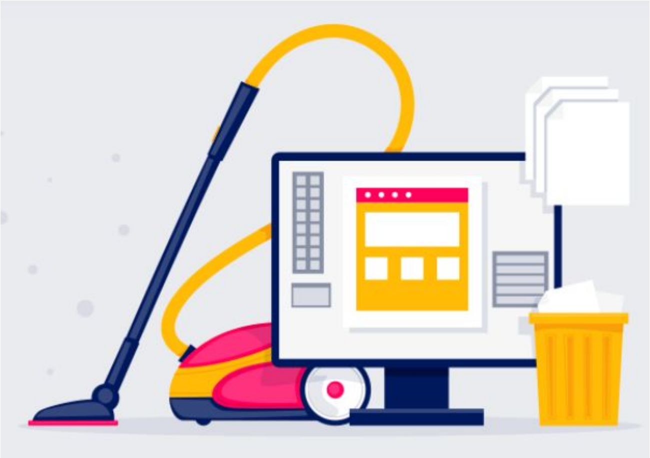
- Load the dataset and examine its structure
- Check for missing values and data types
- Get a feel for the data through basic statistics and visualisations

CoGrammar

Simple EDA Framework

Clean and Preprocess:

- ❖ Handle missing values and outliers
- ❖ Encode categorical variables if necessary
- ❖ Scale or normalise numerical features if required

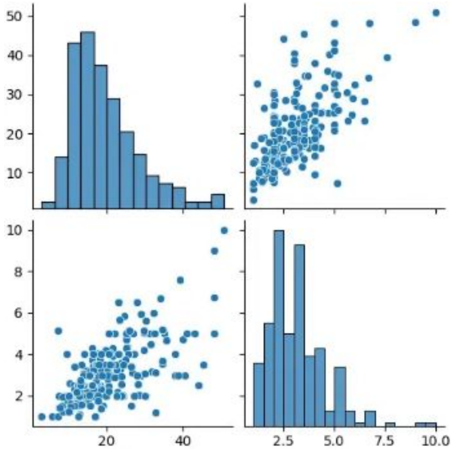


CoGrammar

Simple EDA Framework

Explore Relationships:

- ❖ Analyse relationships between features and the target variable
- ❖ Use visualisations like scatter plots, pair plots, and correlation matrices
- ❖ Identify patterns, trends, and clusters in the data



CoGrammar

Simple EDA Framework

Assess Feature Importance:

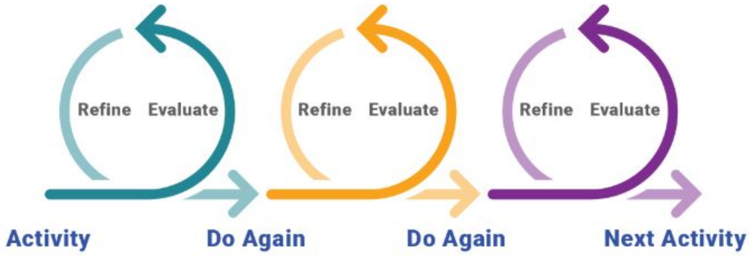
- ❖ Determine the significance of features using statistical tests
- ❖ Use techniques like Decision Trees or Random Forests to evaluate feature importance
- ❖ Select relevant features based on their importance and domain knowledge

CoGrammar

Simple EDA Framework

Iterate and Refine:

- Iterate on the analysis based on the insights gained
- Refine the data cleaning and preprocessing steps if necessary
- Consider additional visualisations or techniques to deepen the understanding of the data

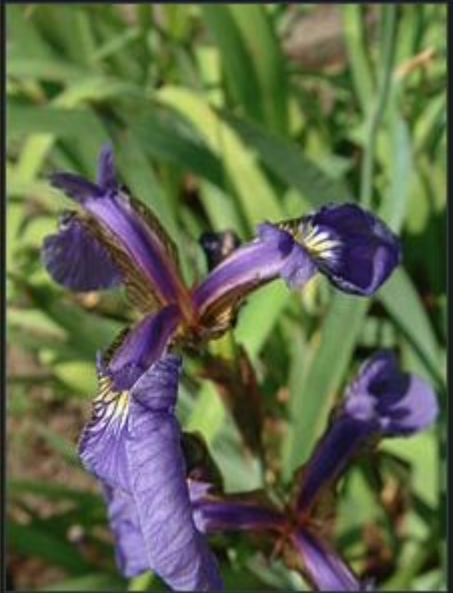


CoGrammar

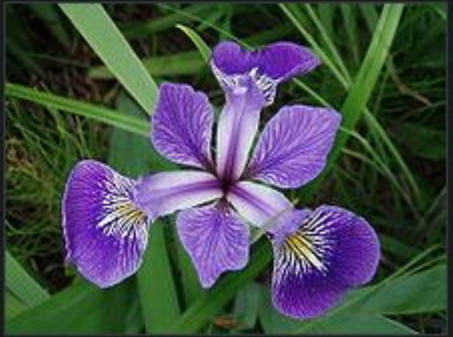
Loading and Exploring the Dataset

Loading and Exploring the Dataset

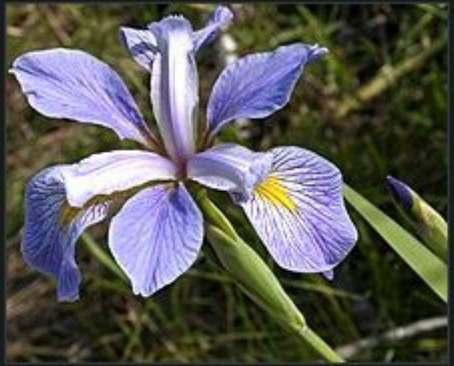
- ❖ To demonstrate EDA techniques, we'll use the Iris dataset from scikit-learn.
- ❖ The Iris dataset consists of measurements of sepal length, sepal width, petal length, and petal width for three species of Iris flowers.
- ❖ We'll load the dataset using scikit-learn and create a pandas DataFrame to work with.



Iris setosa

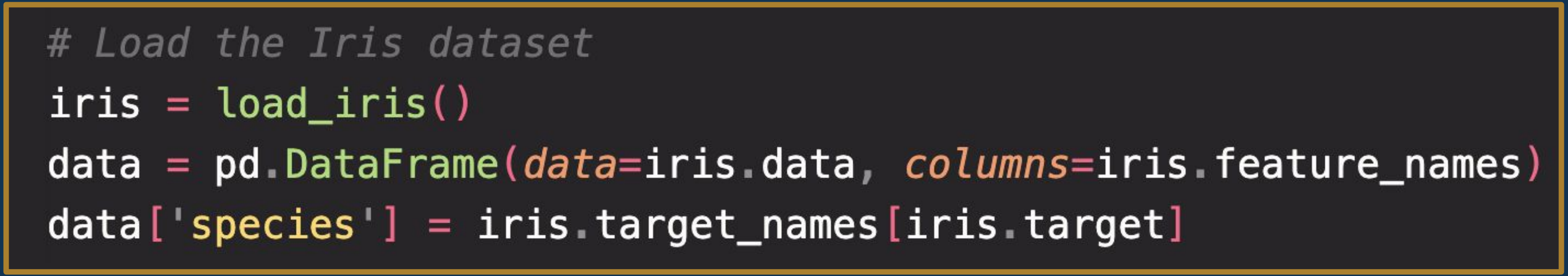


Iris versicolor



Iris virginica

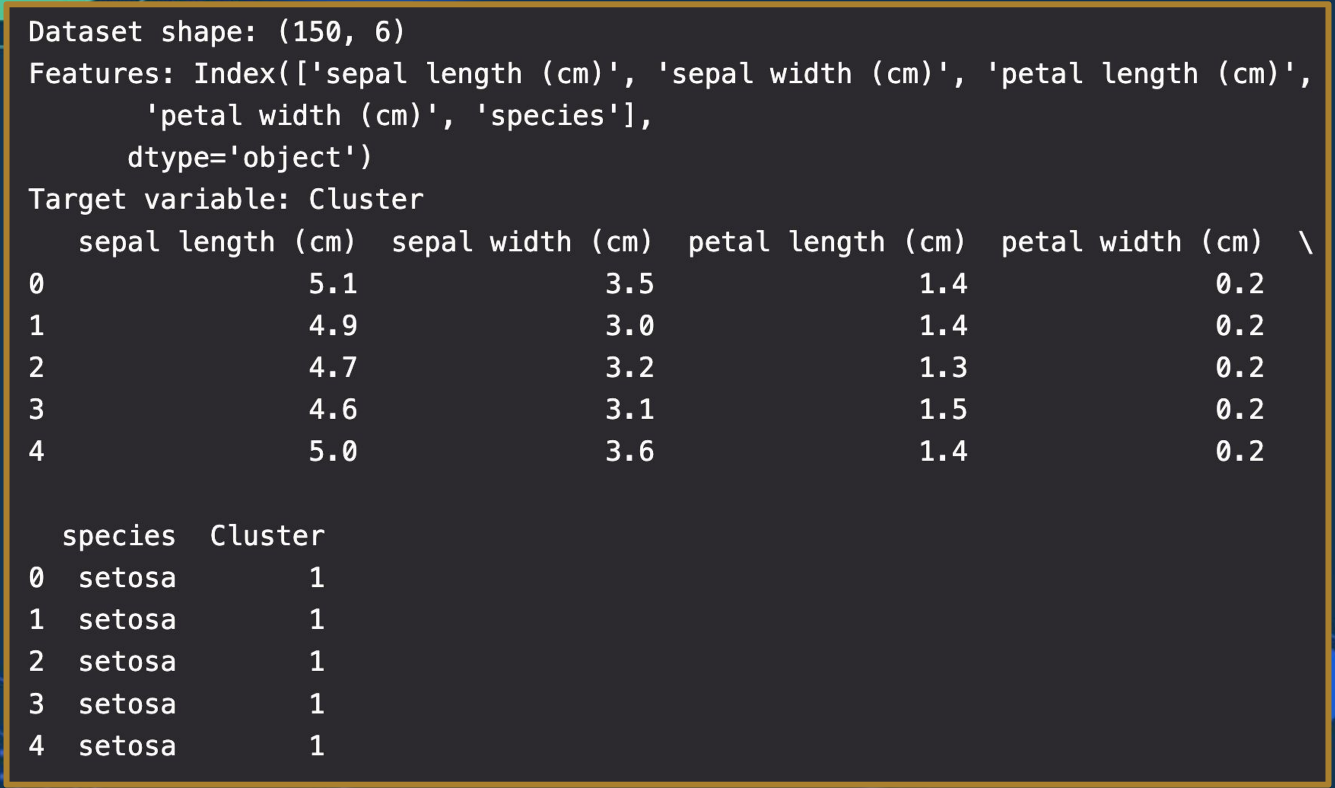
Source: Wikipedia



```python
# Load the Iris dataset
iris = load_iris()
data = pd.DataFrame(data=iris.data, columns=iris.feature_names)
data['species'] = iris.target_names[iris.target]
```

Loading and Exploring the Dataset

- After loading the dataset, we'll explore its basic properties:
  - **Shape of the dataset:** number of rows and columns
  - **Features:** the independent variables in the dataset
  - **Target variable:** the dependent variable (depends on the features) we want to predict or analyse



```
Dataset shape: (150, 6)
Features: Index(['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)',
       'petal width (cm)', 'species'],
      dtype='object')
Target variable: Cluster
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2   
1                4.9               3.0                1.4               0.2   
2                4.7               3.2                1.3               0.2   
3                4.6               3.1                1.5               0.2   
4                5.0               3.6                1.4               0.2   

  species  Cluster  
0  setosa        1  
1  setosa        1  
2  setosa        1  
3  setosa        1  
4  setosa        1  
```

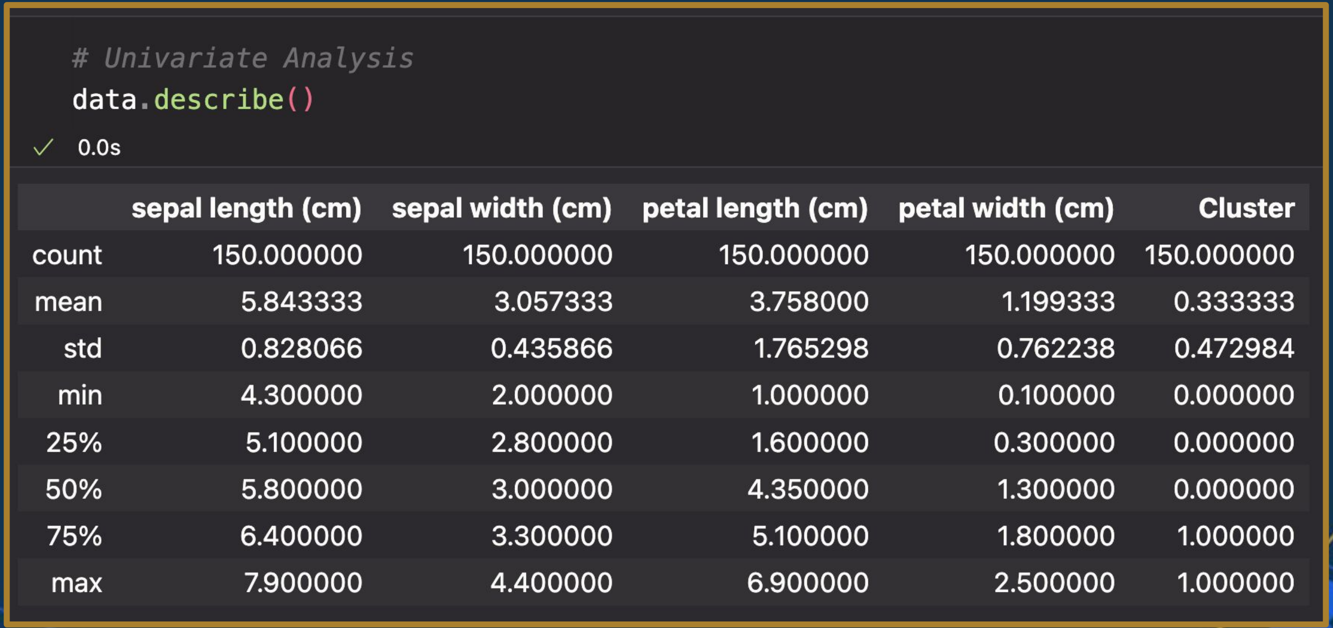
Univariate, Bivariate, and Multivariate Analysis

CoGrammar

Univariate Analysis

- Univariate analysis involves analysing each variable individually.
  - **Univariate:** involving one variate or variable quantity.
- We'll start by calculating descriptive statistics for the numeric variables using the describe() function.
- Descriptive statistics provide a summary of the central tendency, dispersion, and shape of the data.

Univariate Analysis

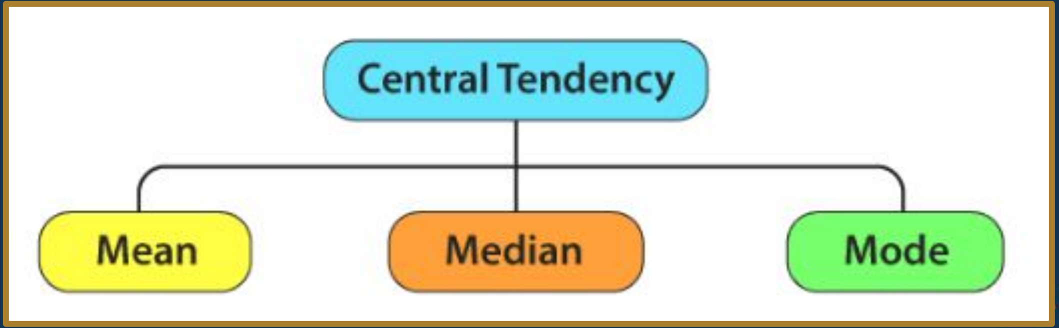


```python
# Univariate Analysis
data.describe()
```

✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Cluster
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333	0.333333
std	0.828066	0.435866	1.765298	0.762238	0.472984
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	0.000000
75%	6.400000	3.300000	5.100000	1.800000	1.000000
max	7.900000	4.400000	6.900000	2.500000	1.000000

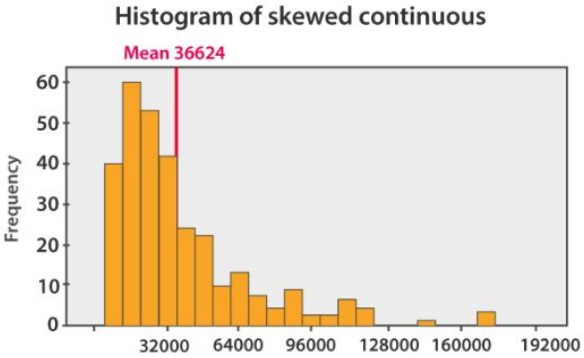
Univariate Analysis: Measures of Central Tendency

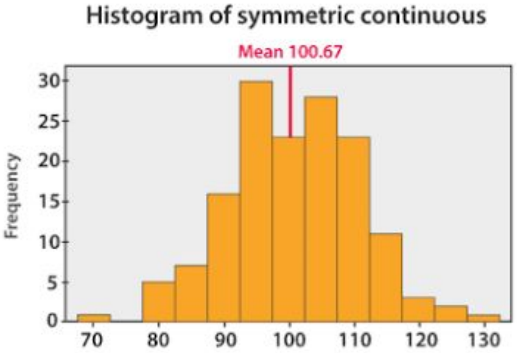


CoGrammar

Measures of Central Tendency

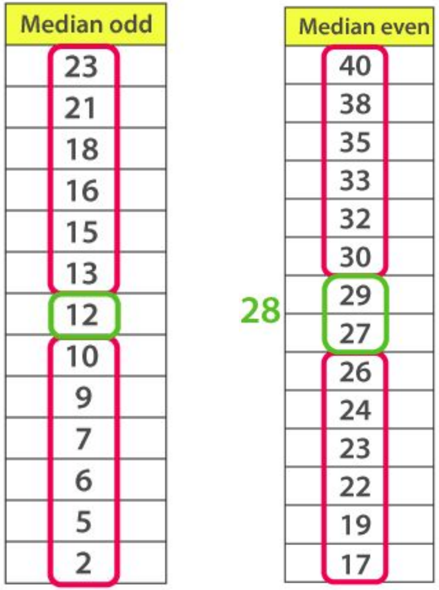
- **Mean** represents the average value of the dataset. It can be calculated as the sum of all the values in the dataset divided by the number of values.





Measures of Central Tendency

- **Median** is the middle value of the dataset in which the dataset is arranged in the ascending order or in descending order. When the dataset contains an even number of values, then the median value of the dataset can be found by taking the mean of the middle two values.



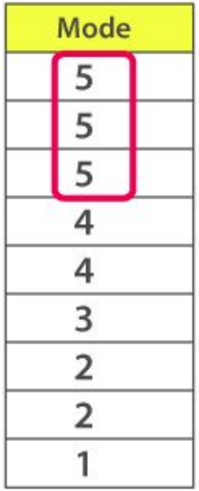
Median odd
23
21
18
16
15
13
12
10
9
7
6
5
2

Median even
40
38
35
33
32
30
29
27
26
24
23
22
19
17

28

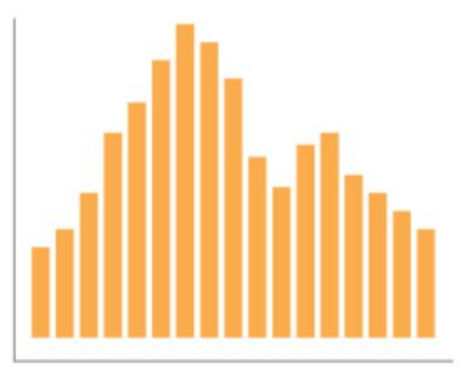
Measures of Central Tendency

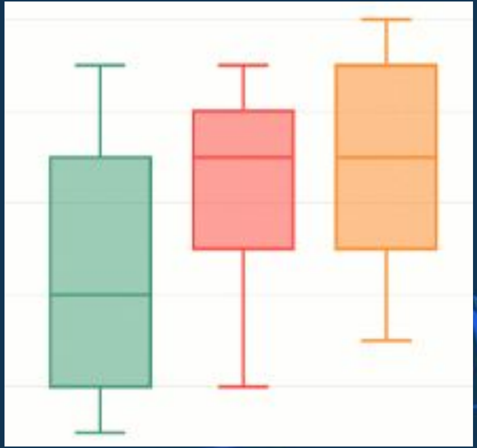
- **Mode** represents the frequently occurring value in the dataset. Sometimes the dataset may contain multiple modes and in some cases, it does not contain any mode at all.



Mode
5
5
5
4
4
3
2
2
1

Univariate Analysis: Distribution Visualisation



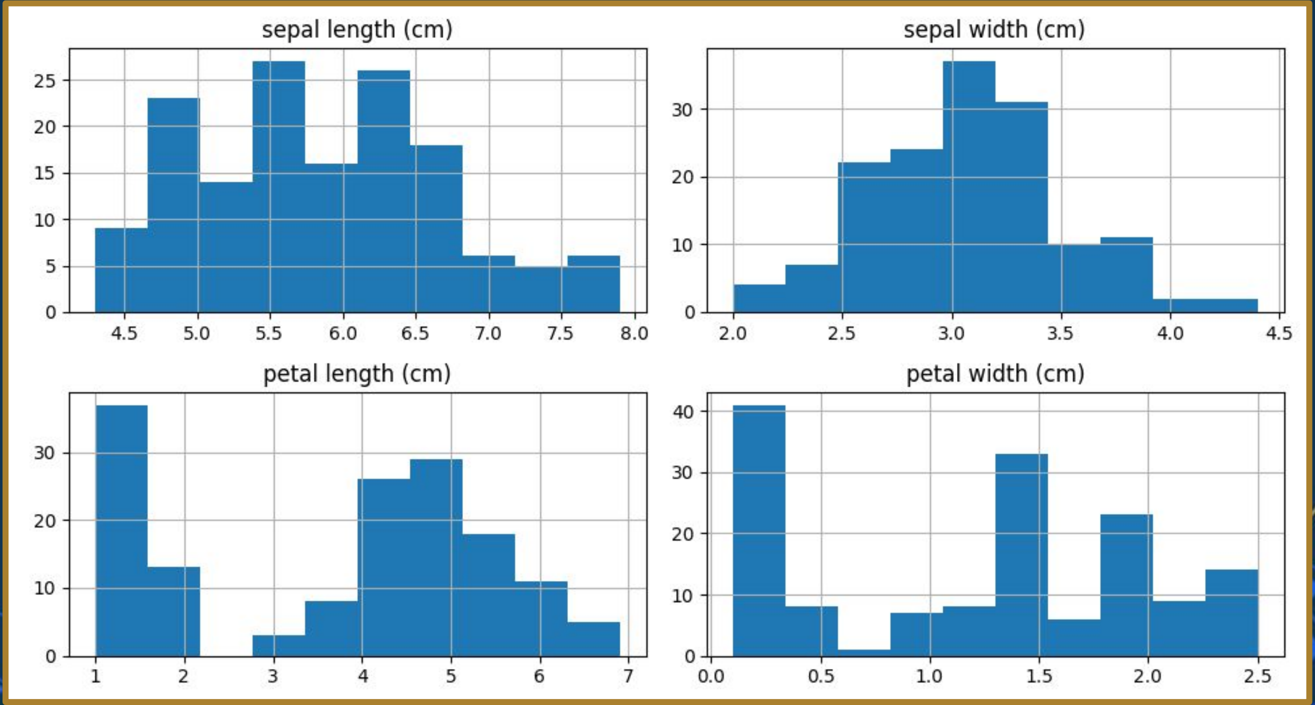


CoGrammar

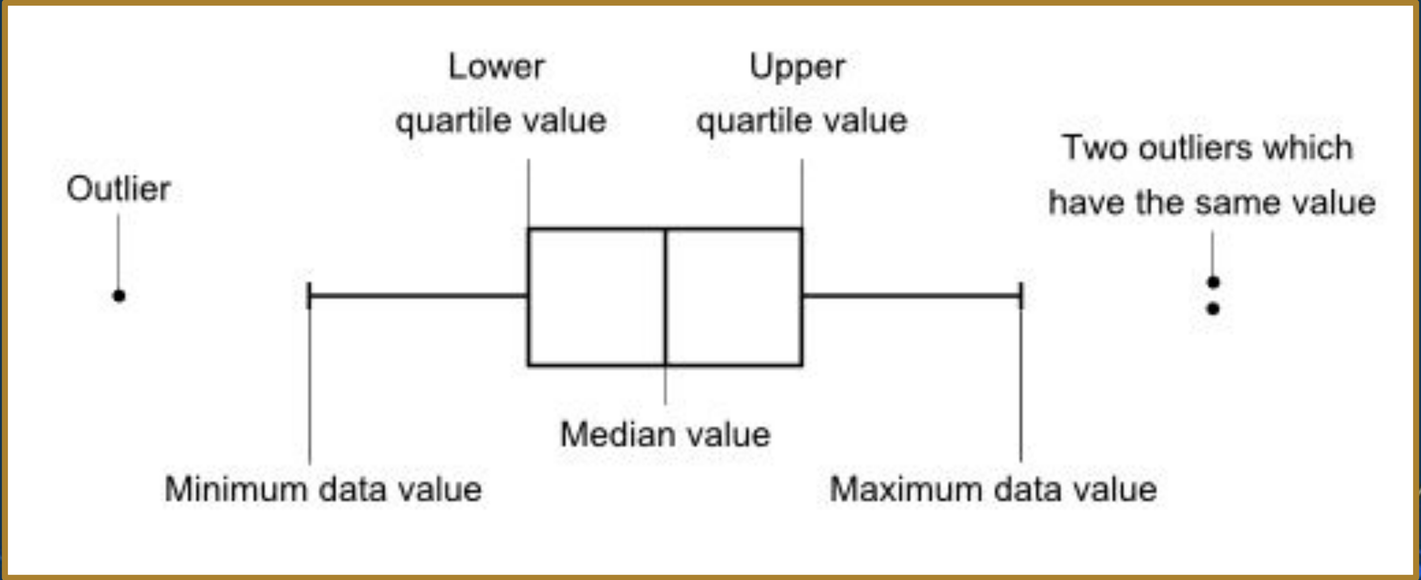
Distribution Visualisation

- Next, we'll visualise the distribution of each feature using histograms and box plots.
- **Histograms** show the frequency distribution of a variable, helping to identify the shape, central tendency, and spread of the data.
- **Box plots** provide a summary of the distribution, highlighting the median, quartiles, and outliers.

Histogram: Iris Dataset Features

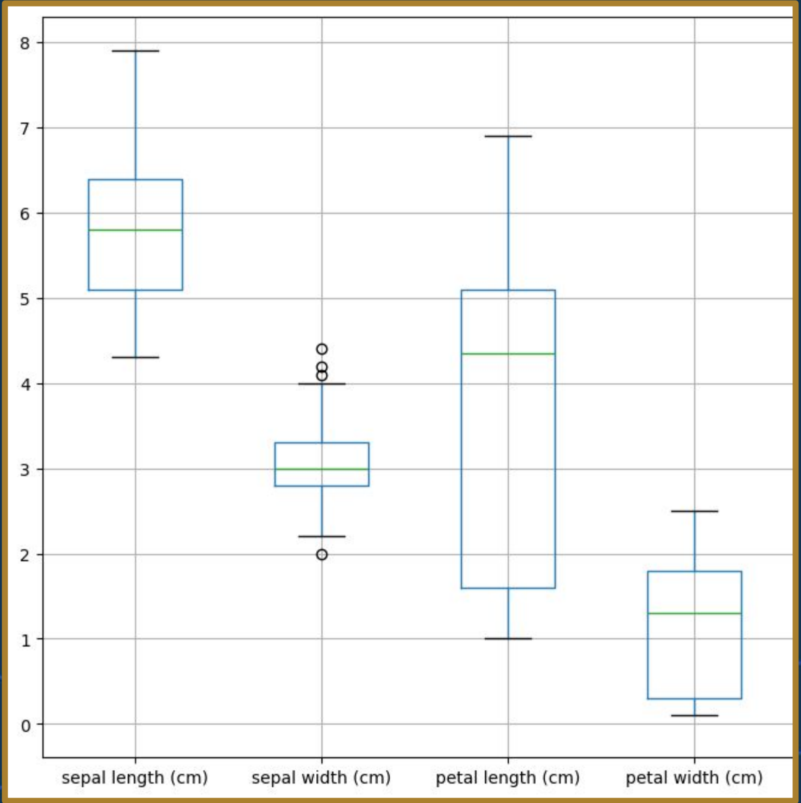


Box Plots



CoGrammar

Box Plots: Iris Dataset Features

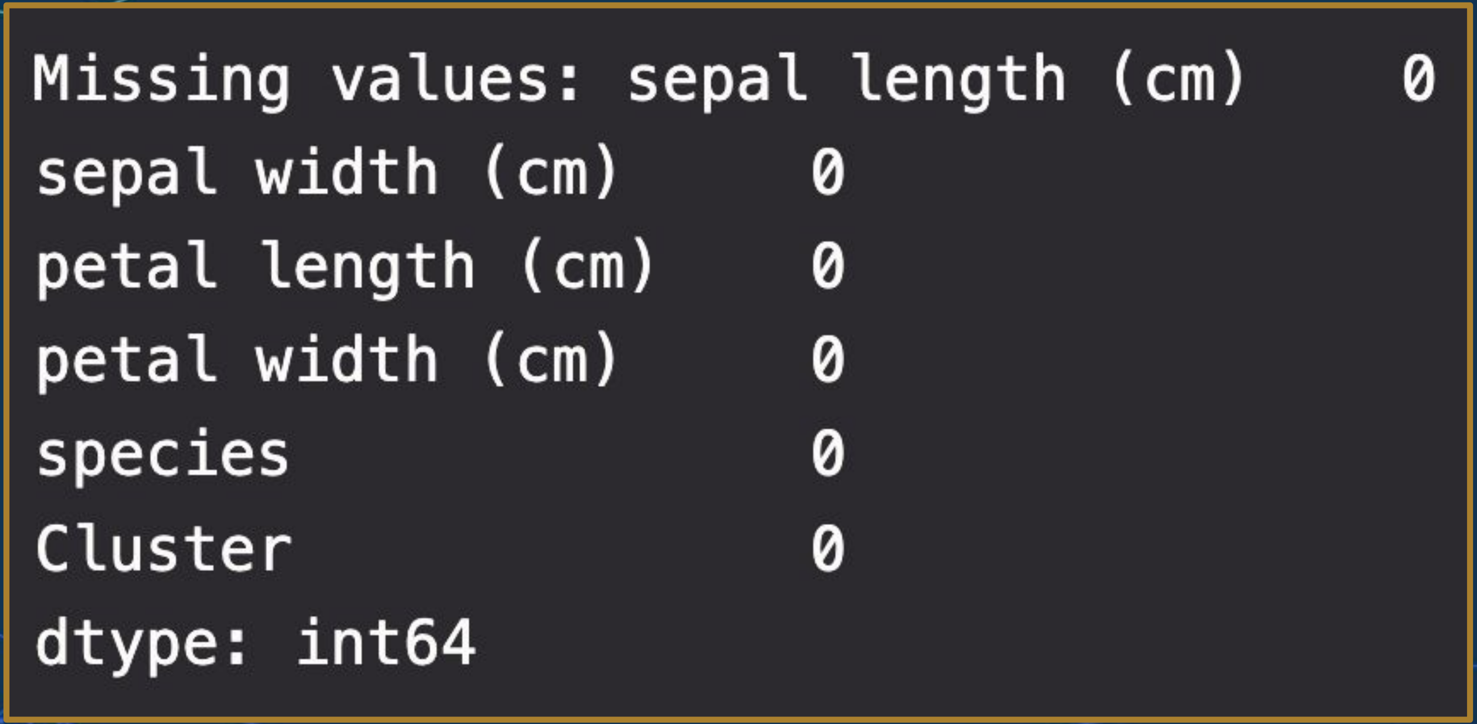


CoGrammar

Univariate Analysis: Missing Values

Missing Values

- We'll also check for missing values in the dataset using the isnull().sum() function.
- Missing values can impact the analysis and need to be handled appropriately.
  - Common strategies include filling missing values with the mean, median, or mode.
  - It’s usually not a good idea to just drop data, as this could skew the data and thus the results of prediction.



```
Missing values: sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
species              0
Cluster              0
dtype: int64
```

Let's take a break

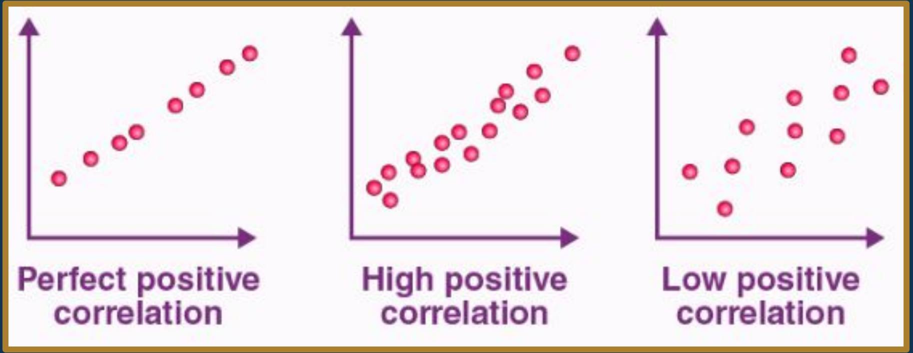
Bivariate Analysis

CoGrammar

Bivariate Analysis

- Bivariate analysis involves examining the relationship between two variables.
  - **Bivariate:** involving or depending on two variates.
- We'll use **scatter plots** to visualise the relationship between features and the target variable.
- Scatter plots help identify patterns, correlations, and clusters in the data.
- A **pair plot** is a plot of subplots where each subplot represents a bivariate distribution, such as a scatterplot, of two variables in the given dataset.

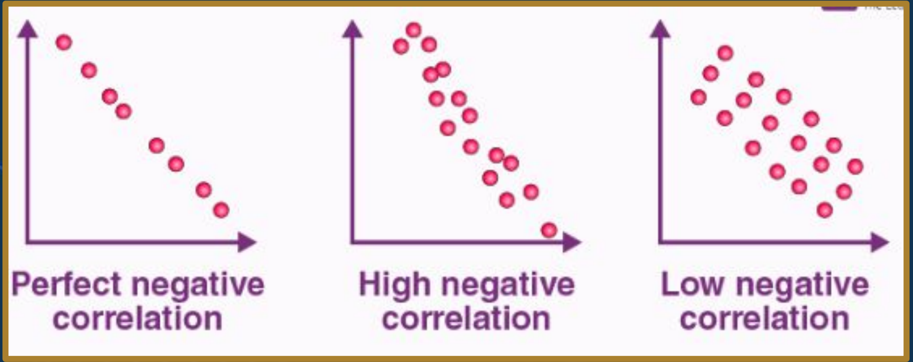
Scatter Plots



Perfect positive correlation

High positive correlation

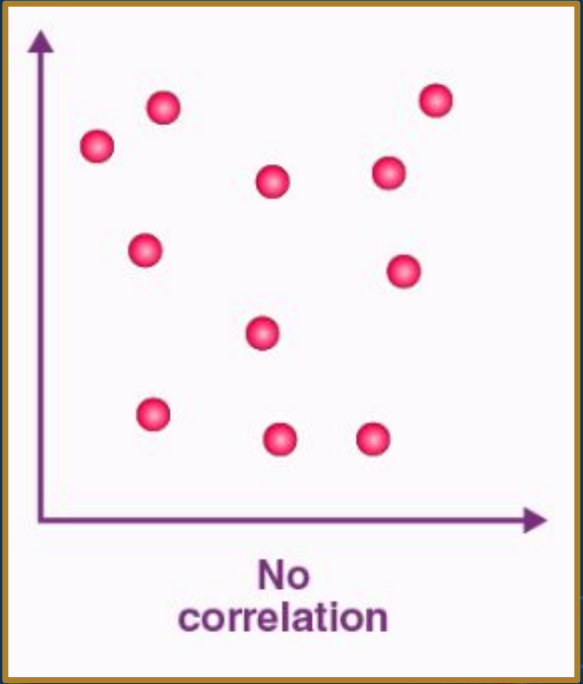
Low positive correlation



Perfect negative correlation

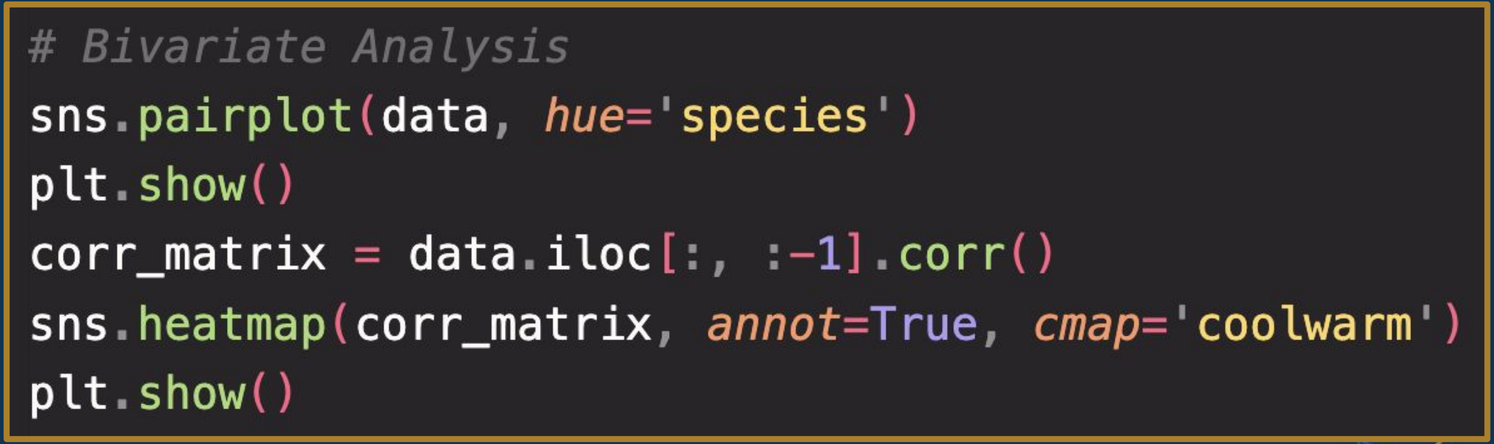
High negative correlation

Low negative correlation



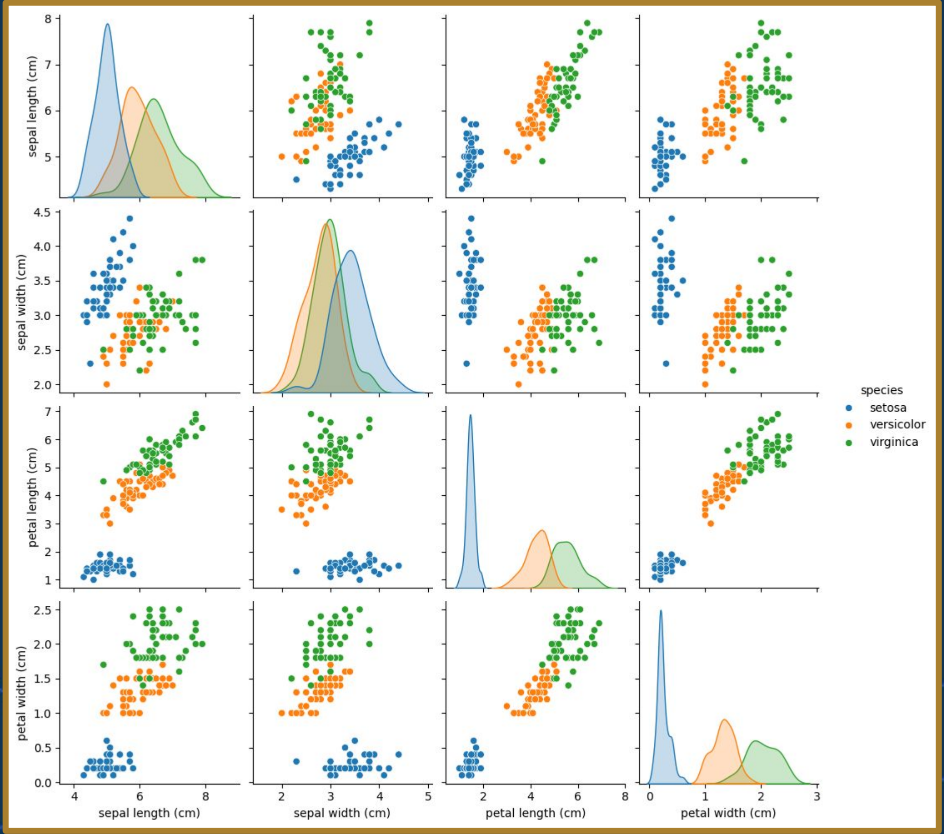
No correlation

Iris Dataset Pairplot



```
# Bivariate Analysis
sns.pairplot(data, hue='species')
plt.show()
corr_matrix = data.iloc[:, :-1].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.show()
```

Iris Dataset Pairplot

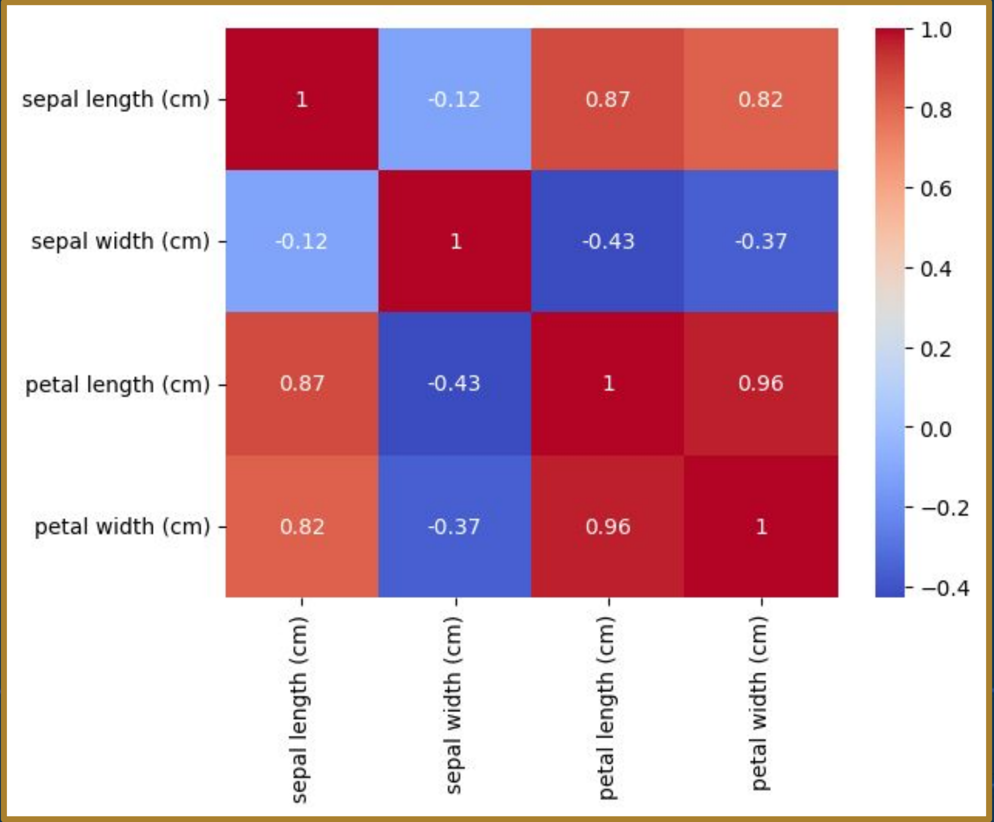


CoGrammar

Bivariate Analysis: Quantifying Relationships

- To **quantify the relationship between numeric features**, we'll calculate the **correlation matrix.**
- The correlation matrix shows the pairwise correlation coefficients between variables.
- We'll visualise the correlation matrix using a **heatmap**.

Heatmap



CoGrammar

Bivariate Analysis: Quantifying Relationships

- ❖ **Interpreting the correlation matrix:**
  - ➢ Correlation coefficients range from -1 to 1.
  - ➢ The high positive correlations between Petal Length and Petal Width (0.96) and between Sepal Length and Petal Length (0.87) suggest that these pairs of features are strongly related and may provide similar information.
  - ➢ The low correlations between Sepal Width and the other features indicate that Sepal Width provides relatively independent information compared to the other features.

What does a scatter plot help identify in bivariate analysis?

1. Frequency distribution of a variable
2. Patterns, correlations, and clusters in the data
3. Median, quartiles, and outliers
4. Missing values in the dataset

What does a scatter plot help identify in bivariate analysis?

1. Frequency distribution of a variable
2. **Patterns, correlations, and clusters in the data**
3. Median, quartiles, and outliers
4. Missing values in the dataset

What does a correlation coefficient of 1 indicate?

1. Perfect positive correlation
2. Perfect negative correlation
3. No correlation
4. Missing data

CoGrammar

What does a correlation coefficient of 1 indicate?

1. **Perfect positive correlation**
2. Perfect negative correlation
3. No correlation
4. Missing data

What does the color intensity in a correlation matrix heatmap represent?

1. The number of variables
2. The strength and direction of correlations
3. The size of the dataset
4. The number of missing values

What does the color intensity in a correlation matrix heatmap represent?

1. The number of variables
2. **The strength and direction of correlations**
3. The size of the dataset
4. The number of missing values

Multivariate Analysis

CoGrammar

Multivariate Analysis

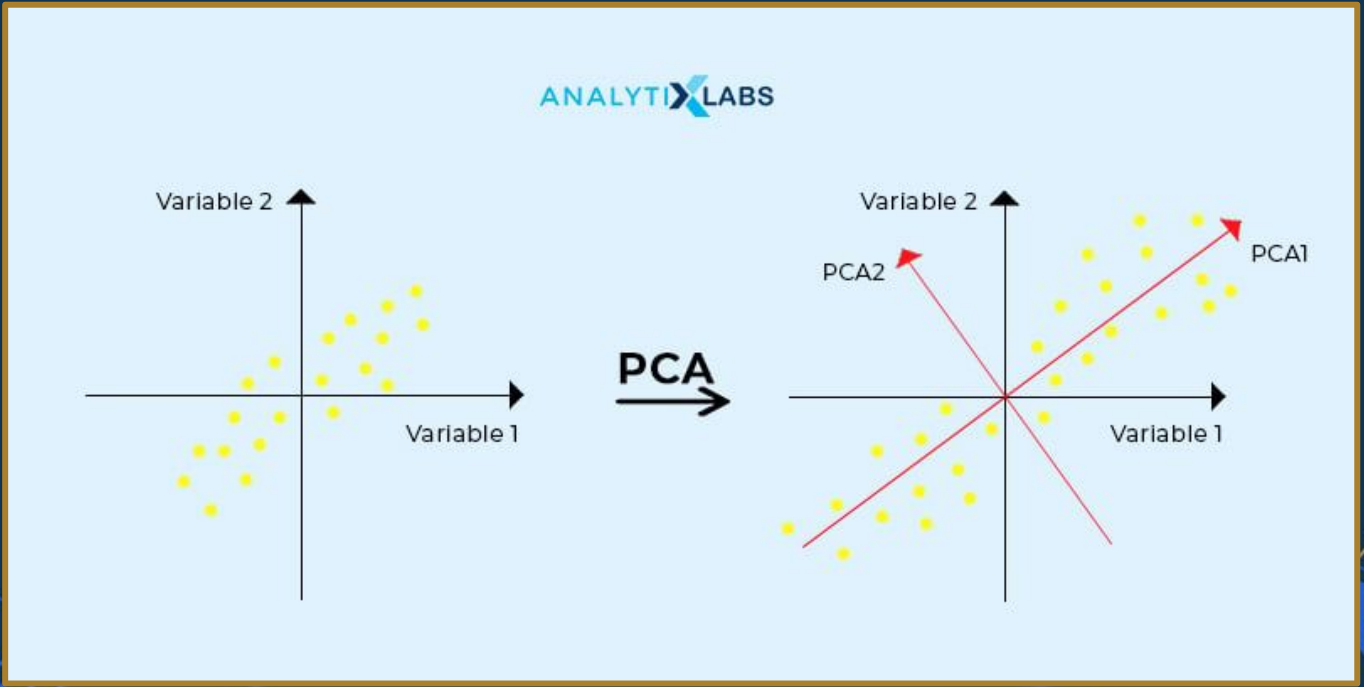
- **Multivariate analysis** involves examining relationships among **multiple variables simultaneously.**
- We’ll look at 3 multivariate analysis techniques:
  - Principal Component Analysis
  - K-means Clustering
  - Feature Importance

Multivariate Analysis:
Principal Component Analysis

Principal Component Analysis

- **Principal Component Analysis (PCA)** is a dimensionality reduction technique that transforms the original features into a new set of uncorrelated features called **principal components.**
- PCA helps **identify patterns and structure in high-dimensional data** by finding the directions of maximum variance.

PCA



Source: [AnalytixLabs]

CoGrammar

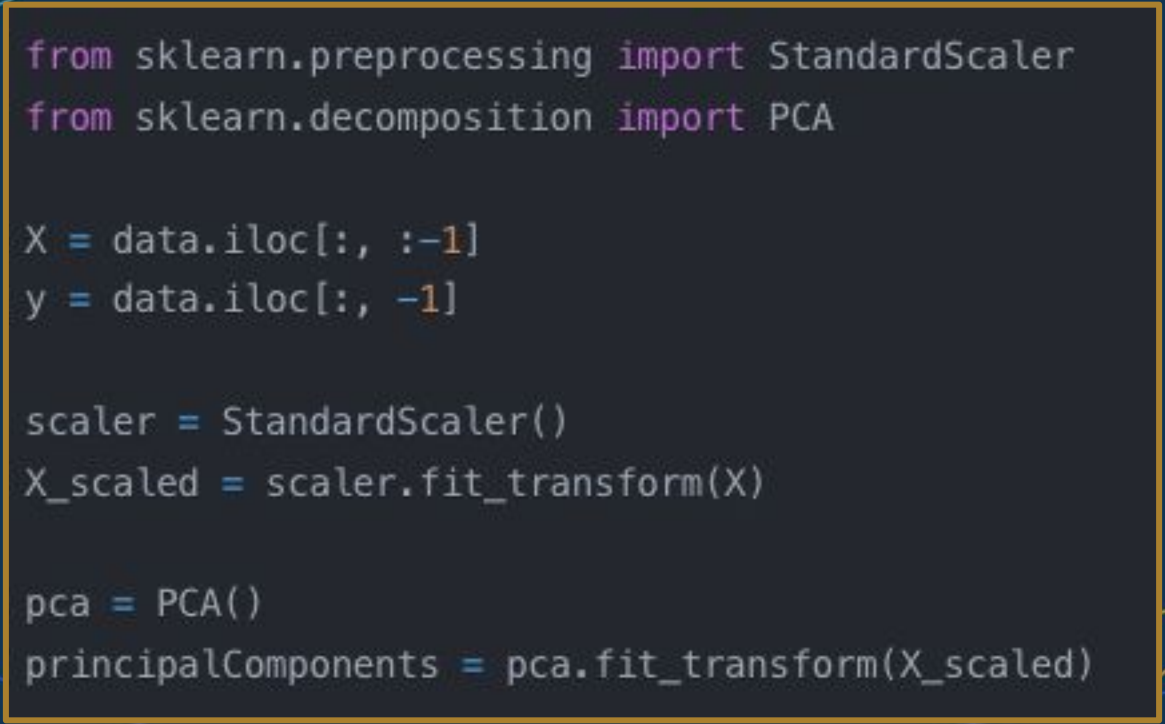
How Is PCA Useful?

There are many advantages and practical uses of PCA in multivariate analysis, including:

1. **Dimensionality reduction:** sometimes we can have datasets that have many variables. Using PCA, we can reduce the number of variables to work with while preserving most of the important information.
2. **Noise reduction:** PCA can filter out random noise and irrelevant variations in the data.
3. **Visualisation of high-dimensional data:** visualising high-dimensional data can be tricky - can you imagine a 10-D scatterplot? PCA makes it possible to possible to visualise complex datasets in 2D or 3D for easier interpretation.

PCA Steps (Python abstracts all the math)

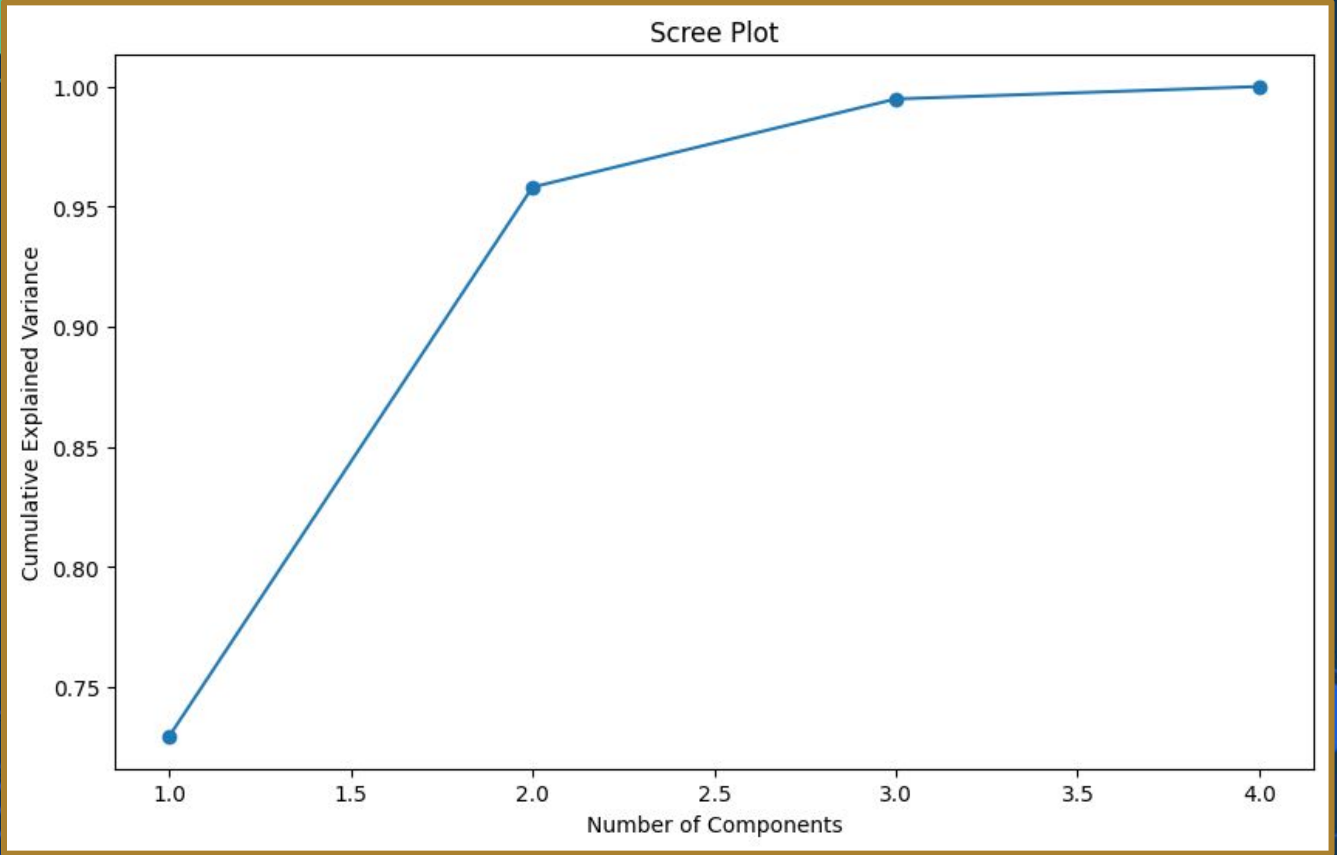
1. Standardise the data to ensure all features have zero mean and unit variance.
2. Compute the covariance matrix of the standardized data.
3. Calculate the eigenvectors and eigenvalues of the covariance matrix.
4. Sort the eigenvectors in descending order of their corresponding eigenvalues.
5. Select the top k eigenvectors as the principal components.
6. Transform the original data into the new feature space defined by the principal components.



```python
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

X = data.iloc[:, :-1]
y = data.iloc[:, -1]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

pca = PCA()
principalComponents = pca.fit_transform(X_scaled)
```

Scree Plots

- To determine the number of principal components to retain, we can analyse the explained variance ratio.
- The explained variance ratio represents the proportion of variance explained by each principal component.
- We can visualise the cumulative explained variance using a scree plot.


Scree Plot
Cumulative Explained Variance
1.00
0.95
0.90
0.85
0.80
0.75
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Number of Components

Scree Plot

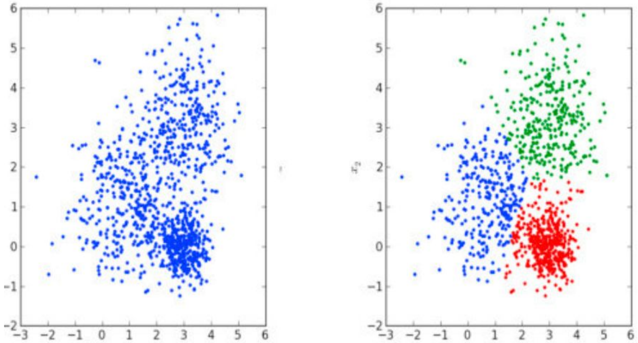
- Interpreting the scree plot:
  - Look for an elbow point where the cumulative explained variance starts to plateau.
  - Choose the number of components that capture a significant portion of the total variance (e.g., 80-90%).
  - In our case it seems to be 2 components.

Multivariate Analysis: K-Means Clustering

CoGrammar

K-Means Clustering

- K-means clustering is an unsupervised learning algorithm that partitions the data into K clusters based on similarity.
- It aims to minimise the within-cluster sum of squares (WCSS) or the Euclidean distance between data points and their cluster centroids.
- The image belows shows a dataset before and after clustering. The result is 3 distinct clusters of similar data points, which helps to identify natural groupings in data, providing insights before applying more complex models.



K-Means Clustering Steps (Python abstracts all the math)

1. Choose the number of clusters K.
2. Initialize K cluster centroids randomly.
3. Assign each data point to the nearest centroid based on Euclidean distance.
4. Update the cluster centroids by taking the mean of the data points assigned to each cluster.
5. Repeat steps 3 and 4 until convergence or a maximum number of iterations is reached.

Multivariate Analysis: Feature Importance

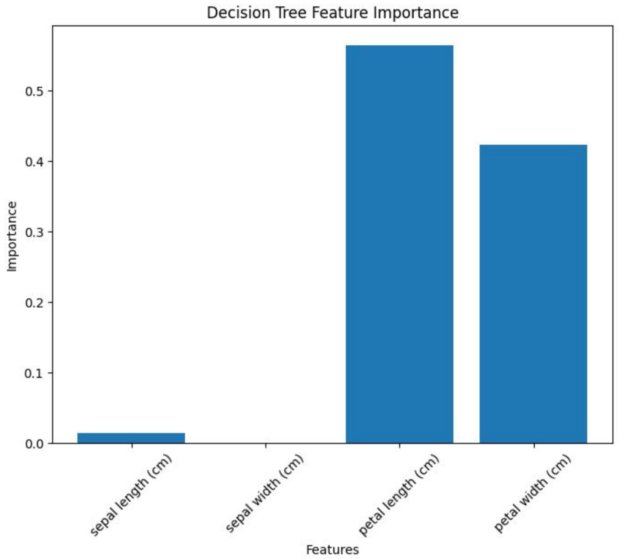
CoGrammar

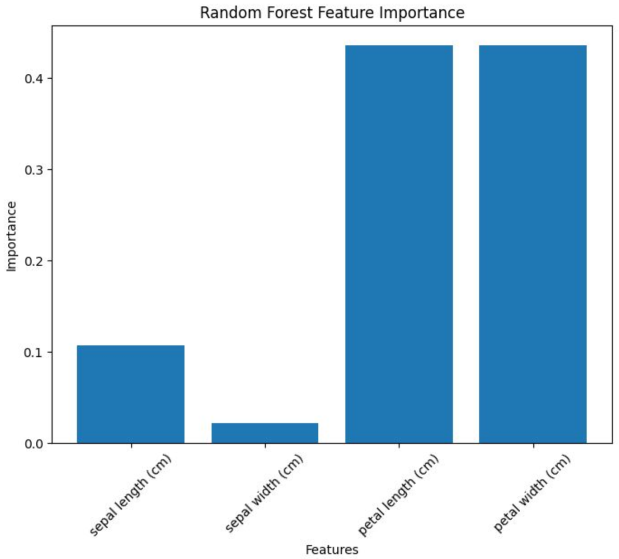
Feature Importance

- **Feature importance** refers to the relative contribution of each feature in predicting the target variable.
- The prediction of a model is only as good as the features used to make the prediction, thus we want the **most important predictors**.
- We'll assess feature importance using statistical tests and machine learning techniques.
- **Decision Trees** and **Random Forests** are machine learning algorithms that can provide **feature importance scores.**
- The importance score represents the decrease in impurity or increase in information gain achieved by splitting on a particular feature.

Feature Importance Plots

The feature importance in the **Decision Tree model** highlights **"petal length" as the most significant feature**, followed by "petal width," while **"sepal length" and "sepal width" contribute minimally.** The **Random Forest model** assigns nearly **equal importance to "petal length" and "petal width,"** with "sepal length" and "sepal width" being **relatively less influential.**





CoGrammar

What is the purpose of Principal Component Analysis (PCA)?

1. To partition the data into clusters
2. To test the significance of the difference in means between groups
3. To transform original features into a new set of uncorrelated features
4. To handle missing values in the dataset

CoGrammar

What is the purpose of Principal Component Analysis (PCA)?

1. To partition the data into clusters
2. To test the significance of the difference in means between groups
3. **To transform original features into a new set of uncorrelated features**
4. To handle missing values in the dataset

In the K-means clustering algorithm, what does the "K" represent?

1. The number of iterations
2. The number of clusters
3. The number of features
4. The number of data points

In the K-means clustering algorithm, what does the "K" represent?

1. The number of iterations
2. **The number of clusters**
3. The number of features
4. The number of data points

Questions and Answers

CoGrammar

Thank you for attending

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education