**Exploratory Data Analysis**

In this module, you will learn what is meant by exploratory data analysis, and you will learn how to perform computations on the data to calculate basic descriptive statistical information, such as mean, median, mode, and quartile values, and use that information to better understand the distribution of the data. You will learn about putting your data into groups to help you visualize the data better, you will learn how to use the Pearson correlation method to compare two continuous numerical variables, and you will learn how to use the Chi-square test to find the association between two categorical variables and how to interpret them.

**Learning Objectives**

* Implement descriptive statistics
* Demonstrate the basics of grouping
* Describe data correlation processes

# **EDA(Exploratory Data Analysis)**

## **Exploratory Data Analysis**

* **Exploratory Data Analysis (EDA)**: A method to analyze data to summarize its main characteristics, understand the dataset, uncover relationships between variables, and identify important variables for problem-solving.
* **Key Question**: What characteristics have the most impact on car prices?
* **Techniques Covered**:
  + **Descriptive Statistics**: Summarizes basic features of a dataset.
  + **Grouping Data**: Using GroupBy to transform datasets.
  + **ANOVA (Analysis of Variance)**: A statistical method to divide variation in observations into distinct components.
  + **Correlation**: Understanding relationships between different variables.
  + **Advanced Correlation**: Introduction to Pearson correlation and correlation heatmaps.

## **Descriptive Statistics**

1. **Descriptive Statistics**:
   * Helps summarize and describe the basic features of a dataset.
   * Provides a short summary about the sample and measures of the data.
2. **Using describe() Function in Pandas**:
   * Automatically computes basic statistics for all numerical variables in a DataFrame.
   * Outputs include:
     + Mean
     + Count of data points
     + Standard deviation
     + Quartiles (25th, 50th, 75th percentiles)
     + Extreme values
   * NaN values are skipped in these statistics.
3. **Categorical Variables**:
   * Variables that can be divided into categories (e.g., drive systems).
   * Use value\_counts() to summarize categorical data.
4. **Box Plots**:
   * Visualize numeric data distributions.
   * Show median, quartiles, inter-quartile range, and outliers.
   * Useful for comparing distributions between groups.
5. **Scatter Plots**:
   * Visualize the relationship between two continuous variables.
   * The predictor variable is plotted on the x-axis, and the target variable on the y-axis.
   * Helps identify relationships, such as positive linear relationships.

Code Examples

1. Using describe() Function

import pandas as pd

# Sample DataFrame

data = {

'price': [20000, 25000, 30000, 15000, 40000],

'engine\_size': [1.6, 2.0, 2.5, 1.4, 3.0]

}

df = pd.DataFrame(data)

# Descriptive statistics

stats = df.describe()

print(stats)

2. Summarizing Categorical Data

# Sample DataFrame with categorical variable

data\_categorical = {

'drive\_system': ['FWD', 'RWD', 'FWD', '4WD', 'RWD', 'FWD']

}

df\_categorical = pd.DataFrame(data\_categorical)

# Summarizing categorical data

category\_summary = df\_categorical['drive\_system'].value\_counts()

print(category\_summary)

3. Creating a Box Plot

import matplotlib.pyplot as plt

# Box plot for price

plt.boxplot(df['price'])

plt.title('Box Plot of Car Prices')

plt.ylabel('Price')

plt.show()

4. Creating a Scatter Plot

# Scatter plot for engine size vs price

plt.scatter(df['engine\_size'], df['price'])

plt.title('Engine Size vs Price')

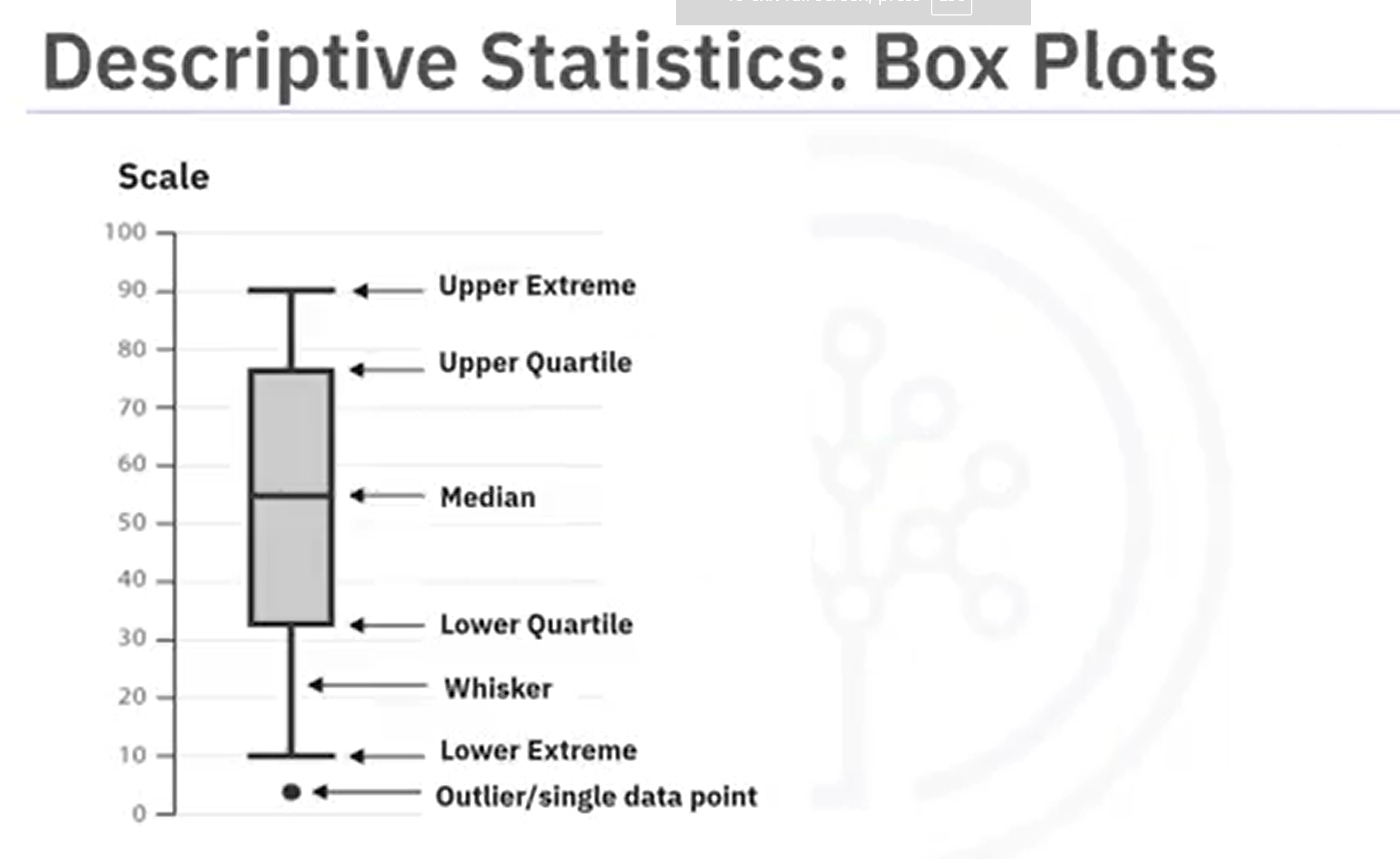
plt.xlabel('Engine Size')

plt.ylabel('Price')

plt.show()

Explanation of Code

* **Descriptive Statistics**: The describe() function provides a summary of the numerical columns in the DataFrame, giving insights into the data distribution.
* **Categorical Summary**: The value\_counts() function counts occurrences of each category in the specified column, helping to understand the distribution of categorical data.
* **Box Plot**: This visual representation helps identify the median, quartiles, and outliers in the price data, making it easier to compare distributions.
* **Scatter Plot**: This plot shows the relationship between engine size and price, allowing you to visually assess if there's a correlation.

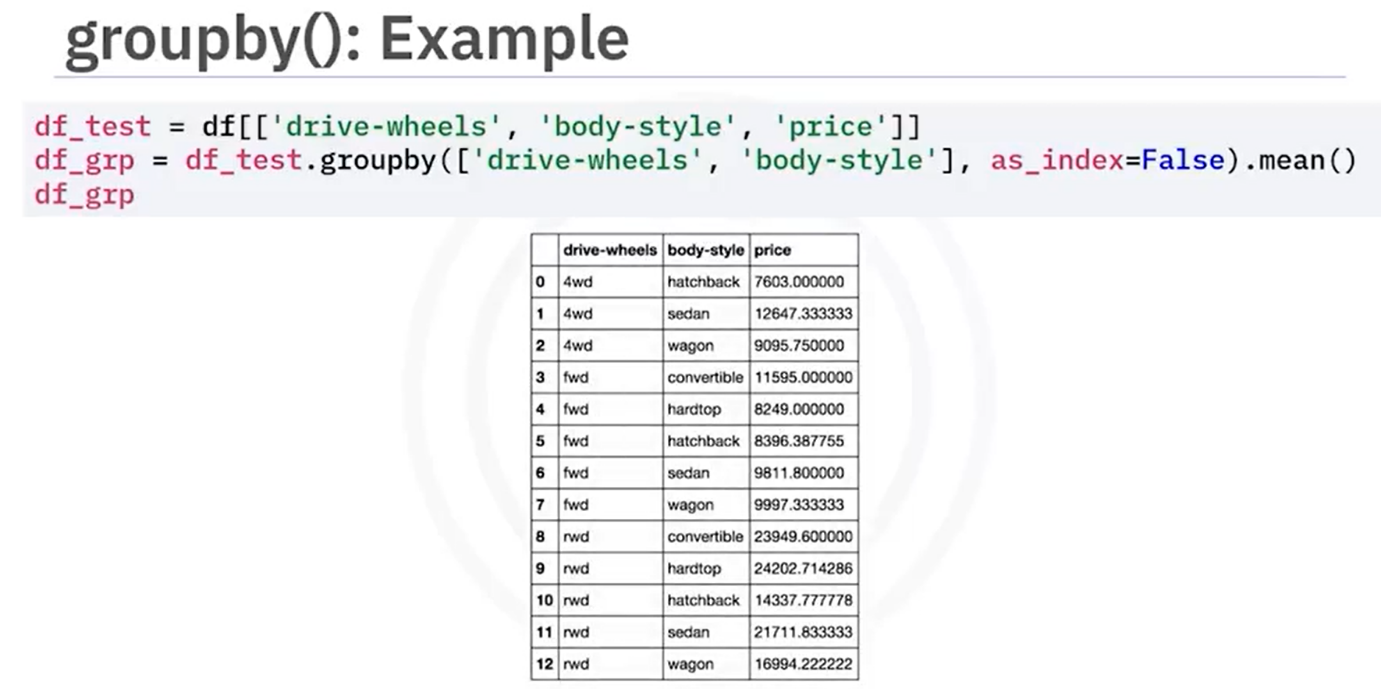


## **GroupBy in Python**

1. **Grouping Data**:
   * The **groupby** method in Pandas is used to group data based on categorical variables.
   * You can group by a single variable or multiple variables.
2. **Example Scenario**:
   * The video discusses analyzing the relationship between different types of drive systems (e.g., forward, rear, and four-wheel drive) and vehicle prices.
3. **Code Example**:
4. # Assuming 'df' is your DataFrame containing vehicle data
5. # Selecting relevant columns
6. reduced\_data = df[['drive\_wheels', 'body\_style', 'price']]
7. # Grouping by drive wheels and body style, then calculating the mean price

grouped\_data = reduced\_data.groupby(['drive\_wheels', 'body\_style']).mean()

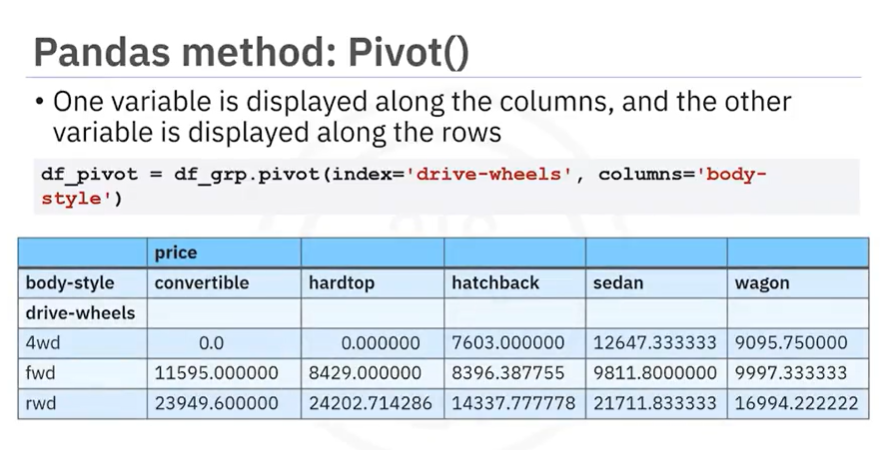
* + **Explanation**:
    - The first line selects the relevant columns from the DataFrame.
    - The second line groups the data by drive\_wheels and body\_style, calculating the average price for each group.



1. **Pivot Tables**:
   * To make the data easier to visualize, you can create a pivot table using the **pivot** method.

pivot\_table = grouped\_data.pivot\_table(index='drive\_wheels', columns='body\_style', values='price')

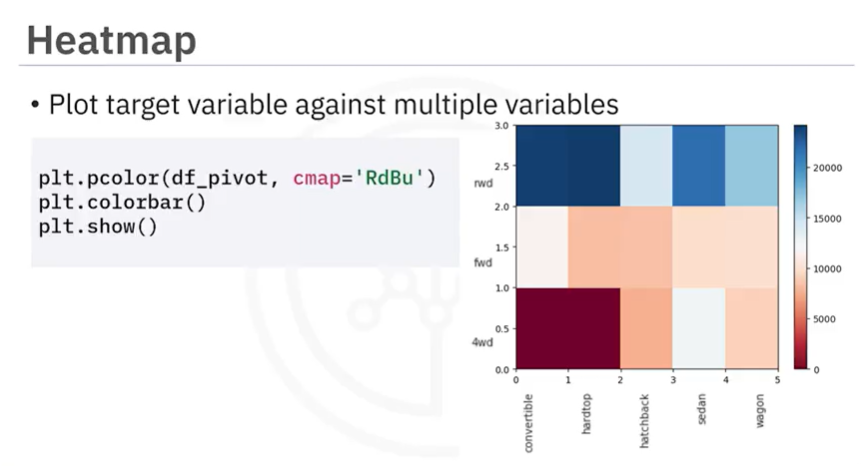
* + **Explanation**:
    - This code transforms the grouped data into a pivot table format, where drive\_wheels are the rows and body\_style are the columns.



1. **Heat Map Visualization**:
   * A heat map can be created to visualize the pivot table data.
2. import matplotlib.pyplot as plt
3. import seaborn as sns
4. # Creating a heat map
5. plt.figure(figsize=(10, 6))
6. sns.heatmap(pivot\_table, annot=True, cmap='RdBu')
7. plt.title('Average Price by Drive Wheels and Body Style')

plt.show()

* + **Explanation**:
    - This code uses Matplotlib and Seaborn to create a heat map, which visually represents the average prices with color intensity.



Summary

* **Grouping** helps in analyzing relationships between categorical variables and numerical values.
* **Pivot tables** provide a clearer view of the grouped data.
* **Heat maps** are effective for visualizing complex data relationships.

## **Correlation**

1. **Correlation**:
   * A statistical metric that measures the extent to which two variables are interdependent.
   * **Correlation does not imply causation**: Just because two variables are correlated does not mean one causes the other.
2. **Examples of Correlation**:
   * **Positive Correlation**: As one variable increases, the other also increases (e.g., engine size and price).
   * **Negative Correlation**: As one variable increases, the other decreases (e.g., highway miles per gallon and price).
   * **Weak Correlation**: Little to no relationship between the variables (e.g., RPM and price).
3. **Visualization**:
   * Scatter plots are used to visualize the relationship between two variables.
   * A regression line can be added to indicate the trend.

Code Examples

Example 1: Positive Correlation (Engine Size and Price)

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Sample data

data = {

'Engine Size': [1.0, 1.5, 2.0, 2.5, 3.0],

'Price': [15000, 18000, 22000, 25000, 30000]

}

df = pd.DataFrame(data)

# Create a scatter plot with a regression line

sns.regplot(x='Engine Size', y='Price', data=df)

plt.title('Positive Correlation between Engine Size and Price')

plt.xlabel('Engine Size (L)')

plt.ylabel('Price (USD)')

plt.show()

**Explanation**:

* This code creates a DataFrame with engine sizes and their corresponding prices.
* The sns.regplot function generates a scatter plot and fits a regression line, showing a positive correlation.

Example 2: Negative Correlation (Highway MPG and Price)

# Sample data

data = {

'Highway MPG': [10, 15, 20, 25, 30],

'Price': [30000, 25000, 20000, 15000, 10000]

}

df = pd.DataFrame(data)

# Create a scatter plot with a regression line

sns.regplot(x='Highway MPG', y='Price', data=df)

plt.title('Negative Correlation between Highway MPG and Price')

plt.xlabel('Highway MPG')

plt.ylabel('Price (USD)')

plt.show()

**Explanation**:

* This code creates a DataFrame with highway miles per gallon and their corresponding prices.
* The scatter plot shows a negative correlation, indicating that as highway MPG increases, the price decreases.

Summary

* Understanding correlation is crucial in data analysis, especially when predicting trends.
* Visualizing data with scatter plots and regression lines helps in identifying relationships between variables.

## **Correlation Statistics**

1. **Pearson Correlation**:
   * A method to measure the strength of the correlation between continuous numerical variables.
   * It provides two values:
     + **Correlation Coefficient**: Indicates the strength and direction of the relationship.
     + **P-value**: Indicates the certainty of the correlation.
2. **Interpreting the Correlation Coefficient**:
   * **Close to 1**: Strong positive correlation.
   * **Close to -1**: Strong negative correlation.
   * **Close to 0**: No correlation.
3. **Interpreting the P-value**:
   * **< 0.001**: Strong certainty about the correlation.
   * **0.001 to 0.05**: Moderate certainty.
   * **0.05 to 0.1**: Weak certainty.
   * **> 0.1**: No certainty of correlation.
4. **Example**:
   * Correlation between **horsepower** and **car price**.
   * If the correlation coefficient is approximately **0.8** and the p-value is much smaller than **0.001**, we conclude a strong positive correlation.
5. **Correlation Heat Map**:
   * Visual representation of the correlation between multiple variables.
   * Diagonal values are always **1** (correlation of a variable with itself).
   * Color scheme indicates the strength of the correlation.

Code Example

Here’s how you can calculate the Pearson Correlation using the **SciPy** library in Python:

import numpy as np

import pandas as pd

from scipy import stats

import seaborn as sns

import matplotlib.pyplot as plt

# Sample data

data = {

'horsepower': [130, 165, 150, 140, 198],

'car\_price': [4000, 5000, 4500, 4800, 6000]

}

# Create a DataFrame

df = pd.DataFrame(data)

# Calculate Pearson Correlation

correlation\_coefficient, p\_value = stats.pearsonr(df['horsepower'], df['car\_price'])

print(f"Correlation Coefficient: {correlation\_coefficient}")

print(f"P-value: {p\_value}")

# Create a heatmap

correlation\_matrix = df.corr()

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm')

plt.title('Correlation Heatmap')

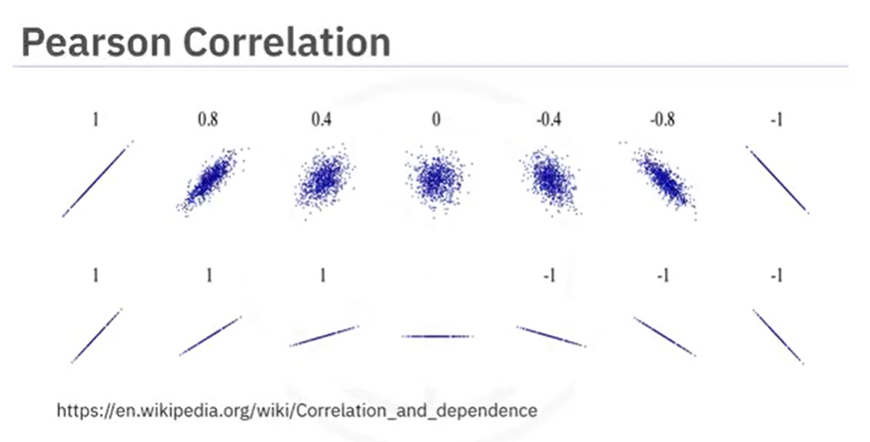
plt.show()

Explanation of the Code

* **Import Libraries**: We import necessary libraries like numpy, pandas, scipy, seaborn, and matplotlib.
* **Sample Data**: We create a sample dataset with horsepower and car\_price.
* **DataFrame**: We convert the data into a pandas DataFrame for easier manipulation.
* **Pearson Correlation Calculation**: We use stats.pearsonr() to calculate the correlation coefficient and p-value.
* **Heatmap**: We create a heatmap to visualize the correlation between the variables.

Conclusion

Understanding Pearson Correlation helps in analyzing relationships between variables effectively. The code provided allows you to calculate and visualize these correlations using Python.



**Lesson Summary**

Congratulations! You have completed this lesson. At this point in the course, you know:

* Tools like the **'describe'** function in pandas can quickly calculate key statistical measures like mean, standard deviation, and quartiles for all numerical variables in your data frame.
* Use the **'value\_counts'** function to summarize data into different categories for categorical data.
* Box plots offer a more visual representation of the data's distribution for numerical data, indicating features like the median, quartiles, and outliers.
* Scatter plots are excellent for exploring relationships between continuous variables, like engine size and price, in a car data set.
* Use Pandas' **'groupby'** method to explore relationships between categorical variables.
* Use pivot tables and heat maps for better data visualizations.
* Correlation between variables is a statistical measure that indicates how the changes in one variable might be associated with changes in another variable.
* When exploring correlation, use scatter plots combined with a regression line to visualize relationships between variables.
* Visualization functions like **regplot,** from the **seaborn** library, are especially useful for exploring correlation.
* The **Pearson correlation**, a key method for assessing the correlation between continuous numerical variables, provides two critical values—the coefficient, which indicates the strength and direction of the correlation, and the P-value, which assesses the certainty of the correlation.
* A correlation coefficient close to 1 or -1 indicates a strong positive or negative correlation, respectively, while one close to zero suggests no correlation.
* For P-values, values less than .001 indicate strong certainty in the correlation, while larger values indicate less certainty. Both the coefficient and P-value are important for confirming a strong correlation.
* Heatmaps provide a comprehensive visual summary of the strength and direction of correlations among multiple variables.