Support Session: Presentation

Agenda

1. Steps involved in a data analysis project
2. Data analysis packages in python - How to import and use them?
3. Common statistical measures - What is the central tendency of different variables? What is the relationship between the variables?
4. Data Visualisation with Python - Why is visualisation important? How to choose plots for Univariate and Bivariate Analysis?

Getting started with Data Analysis

EDA is the approach to explore the data in a systematic way and summarize the main characteristics, using different types of visualizations and analytical tools. Steps involved in a data analysis project using Python are provided below -

1 Importing Packages

In this step, we import all the necessary packages such as **numpy**, **pandas**, **matplotlib**, **seaborn** etc.

2 Loading the Dataset

Using **pandas** functions, we load the dataset in a dataframe. For csv files, '**pd.read_csv( )**' is used. For excel files, 'pd.read_excel( )' is used.

3 Exploratory Data Analysis

In this step, we look for the **shape** of the dataset, the different **data types,** check for **anomalous and missing values,** and analyse the attributes individually as well as relationships between them to identify key business insights

Pandas - Data Analysis packages

Pandas is used for data manipulation and analysis. Some important functions of these packages are provided below -

df.head( )

The **df.head( )** function returns the **first 5 rows** of the dataframe

df.shape

The **df.shape** returns the number of **rows** and **columns** of the dataframe

df.astype( )

The **df.astype( )** function **convert the data type** of an existing column in a dataframe

df.info( )

The **df.info()** function returns information about the dataframe including the **data types** of each column and **memory usage**

df.describe( )

The **df.describe()** function returns the statistical data like percentile, mean, etc. of the dataframe

df.unique( )

The **df.unique()** function returns the unique values present in a dataframe

df.groupby( )

The **df.groupby( )** function function is used to **split** the data **into groups**

df.value_counts( )

The **df.value_counts( )** returns a Series containing the **counts of unique values**.

Common Statistical Measures

Central tendency measures condense the dataset down to one representative value, which is useful for working with large amounts of data. It also allows us to compare one dataset to another.

Measure of Central Tendency

Mean	Median	Mode
The **mean** is the arithmetic average of a set of given numbers.	The **median** is the middle score in a set of given numbers.	The **mode** is the most frequent score in a set of given numbers.
df['column_name'].mean( )	df['column_name'].median( )	df['column_name'].mode( )[0]
The **mean** can be **used to represent the typical value** and therefore serves as a yardstick for all observations.	Since the mean is highly affected by the outliers, the **median** is a better choice for a dataset with extreme values	Mode is the preferred measure when data is categorical.

Common Statistical Measures

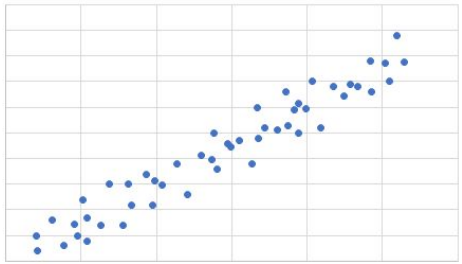
Correlation Coefficient

Correlation is a measure of association between two variables. The **Correlation Coefficient** is a statistical measure of the strength of the relationship between two variables.

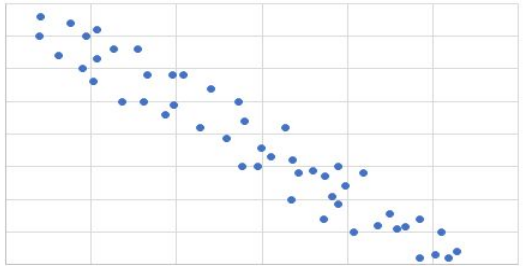
df.corr( )

Based on direction of change in the value of one variable as the value of the other changes, the two variables are said to have a positive relationship, negative relationship, or no relationship at all.

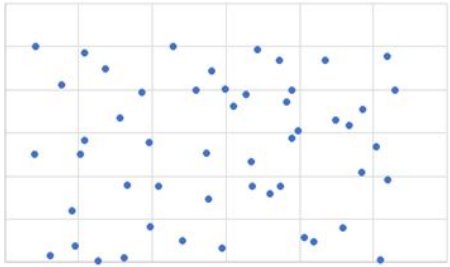
Positive Correlation



Negative Correlation



Zero Correlation



Significance of Data Visualization

Data visualization gives us a **better idea of the information stored in data by giving it visual context through various plots**.

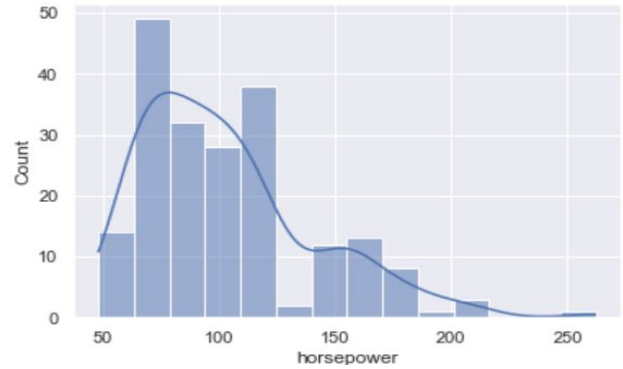
- Using graphic representations, we can visualize large volumes of data in an understandable and coherent way, which in turn helps us comprehend the information and draw conclusions and insights
- Data storytelling is a medium that enables us to easily create a narrative through graphics and diagrams, through which, with the help of visual analytics, we can uncover new insights and engage others.
- It also enables us to **identify relationships and patterns within data,** since discerning trends in the data gives us a competitive advantage

How to choose plots for Univariate Analysis?

When to use a Histogram

When **the data is numeric** and **you want to see the shape of the data distribution**, determine whether the data is distributed approximately normally (bell shaped) or not.

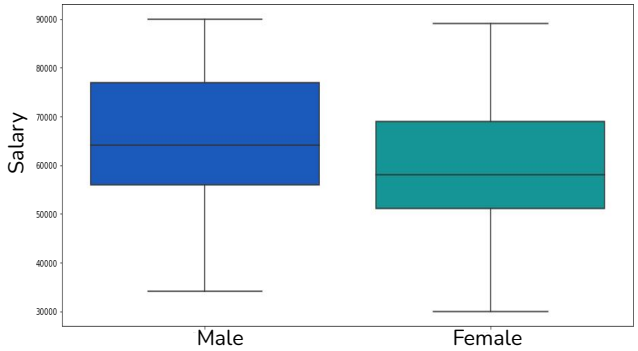
```
sns.histplot( data =  , x = ' ', kde = True )
```



When to use Boxplot

When **the data is numeric** and **you want to compare distributions** across different categorical variables because the centre, spread and overall range are immediately apparent.

```
sns.boxplot( data =  , x = ' ', y = ' ' )
```

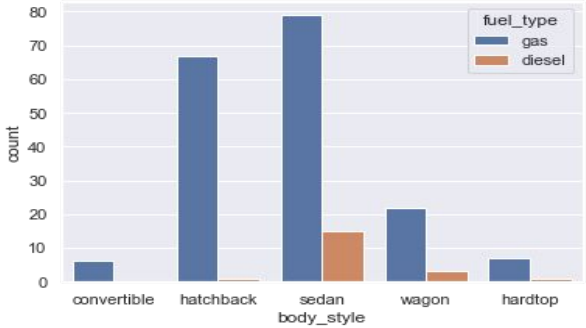


How to choose plots for Categorical variables?

When to use a Count plot

When **the data is categorical** and you want to show the **counts of observations in each categorical** bin.

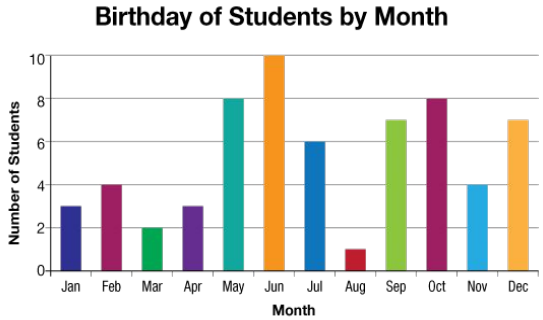
sns.countplot(data = , x = ‘ ‘, y = ‘ ‘)



When to use a Bar plot

When **the data has numeric columns as well as categorical columns** and **you want to see the** the relationship between them or perform a comparison of metric values across different categories of your data.

sns.barplot(data = , x = ‘ ‘, y = ‘ ‘)

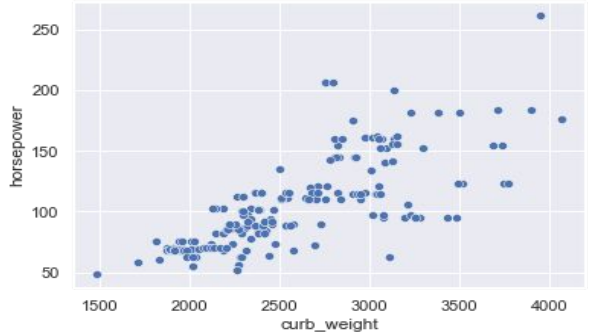


How to choose plots for Bivariate Analysis?

When to use a scatter plot

When **the data is numeric** and **you want to** determine whether the two variables are related, and see if it's a positive or negative correlation.

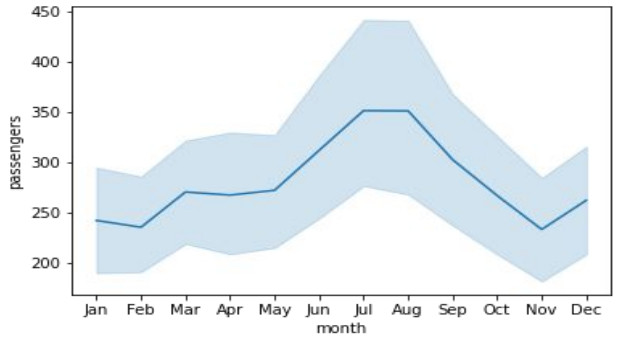
sns.scatterplot(data = , x = ‘ ‘, y = ‘ ‘)



When to use a line chart

When **the data is continuous** and **you want to see the** how the value of something changes over short and long periods of time.

sns.lineplot(data = , x = ‘ ‘, y = ‘ ‘)



greatlearning
Power Ahead

Happy Learning !