DATA EXPLORATION AND VISUALIZATION

A MINI PROJECT REPORT

Submitted by

MITHUN PANDI T	(9517202309070)
KISHORE R	(9517202309058)
VISWA SAGAR P	(95172023090122)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

23AD353 - MINI PROJECT - I

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERIN COLLEGE
SIVAKASI

NOVEMBER 2024

BONAFIDE CERTIFICATE

This is to certify that it is the Bonafide work done by ***MITHUN PANDI T(9517202309070), KISHORE R(9517202309058),VISWASAGAR P (95172023090122)*** *for the course* ***23AD353 - Mini Project - I*** *work entitled "***DATA EXPLORATION AND VISUALIZATION***" at Department of Artificial Intelligence & Data Science, Mepco Schlenk Engineering College, Sivakasi during the year 2024 – 2025.*

……………………………… ………………………………

Faculty In-Charge ***Head of the Department***

Submitted for the Practical Examination held at Mepco Schlenk Engineering College, Sivakasi on ……………………….

…………………………. …………………………

Internal Examiner - I ***Internal Examiner - II***

ACKNOWLEDGEMENT

First and foremost, we express our deepest gratitude to the **LORD ALMIGHTY** for his abundant blessings, which have guided us through our past, present, and future endeavors, making this project a successful reality.

We extend our sincere thanks to our management, esteemed Principal, **Dr. S. Arivazhagan, M.E., Ph.D.**, for providing us with a conducive environment, including advanced systems and well-equipped library facilities, which have been instrumental in the completion of this project.

We are profoundly grateful to our **Head of the Artificial Intelligence and Data Science Department**, **Dr. J. Angela Jennifa Sujana, M.Tech., Ph.D.**, for granting us this golden opportunity to work on a project of this significance and for her invaluable guidance and encouragement throughout.

Our heartfelt appreciation goes to our project guides, **Mrs. A. Muthulakshmi, B.Tech , M.Tech**, Assistant Professor, and **Mrs. K. Kamala Devi, M.E. (Ph.D.)**, Assistant Professor, Department of Artificial Intelligence and Data Science for their unwavering support, insightful suggestions, and expert guidance. Their experience and encouragement have been vital in shaping our project and helping us bring forth our best.

We also extend our sincere gratitude to all our faculty members, lab technicians, and our beloved family and friends, whose timely assistance and moral support played a pivotal role in making this mini-project a success.

SYNOPSIS

MODULE 1: Data exploration involves examining real-world datasets to understand their structure, format, and insights while addressing issues like noise or inconsistencies. It also includes collecting data from various sources, such as APIs, which provide real-time, structured data in formats like JSON or XML for analysis.

MODULE 2: Data processing with NumPy and pandas enables efficient handling, cleaning, and transformation of numerical and structured data. Scikit-learn aids in analyzing individual features, scaling, normalizing, and selecting the most impactful ones for predictive modeling.

MODULE 3: Seaborn and Matplotlib provide robust tools for creating detailed and customizable visualizations to explore data patterns and relationships. AutoViz automates visualization, generating insightful plots to quickly understand trends and anomalies in datasets.

MODULE 4: Python and scikit-learn streamline data cleaning by addressing inconsistencies, imputing missing values, and ensuring data completeness. They also provide efficient normalization techniques to scale features uniformly, enhancing model performance.

MODULE 5: Geospatial visualization tools like GeoPandas help map and analyze location-based patterns. Temporal visualization using Matplotlib or Plotly uncovers trends and relationships in time-series data.

MODULE 6: Python enables efficient data transformation with tools like pandas for scaling, encoding, and reshaping. It also facilitates data wrangling by cleaning and structuring raw data into an analysis-ready format.

MODULE 7: R provides robust tools like ggplot2 and plotly for creating insightful and customizable visualizations. Power BI allows building interactive dashboards, integrating data sources for real-time analytics and decision-making.

TABLE OF CONTENTS

MODULE - 1

DATA EXPLORATION

1.1. Exploration of real – world datasets : Sample Sub heading format:

Data exploration involves examining real-world datasets to understand their structure, features, and characteristics. Real-world datasets often contain incomplete, inconsistent, or noisy data that needs to be cleaned and prepared for analysis. This step helps in identifying patterns, trends, and insights critical for decision-making and model development.

1.2. Analyze Different Formats of Datasets

Datasets are available in various formats such as CSV, JSON, XML, Excel, or database files. Each format has unique features and requires appropriate tools for analysis. For example:

CSV: Simple tabular data stored in plain text, ideal for spreadsheet applications.

JSON/XML: Hierarchical data structures commonly used for APIs and web-based data.

Excel: A versatile format supporting tabular data with multiple sheets.

Database Files: Structured data stored in relational databases like SQL or NoSQL systems.

Understanding these formats ensures efficient handling and processing of datasets for exploration and analysis.

3. Data Collection Using APIs

Application Programming Interfaces (APIs) are a powerful tool for collecting data from various online sources and services. APIs provide a structured way to interact with web applications to request and retrieve data in formats like JSON or XML.

For instance: Weather APIs: Fetch real-time weather data.

Social Media APIs: Extract trends, posts, or user statistics.

Financial APIs: Access stock market data or currency exchange rates.

Using APIs for data collection allows real-time and automated data retrieval, making it an essential step in modern data science workflows.

4.Ways to Include files into a Dataframe

1. pd.read_csv() : The most common and straightforward way to read a CSV file into a DataFrame

```
import pandas as pd
df = pd.read_csv('file.csv')
```

2. Loading from URLs or APIs : Pandas supports reading CSV files from a URL

```
df = pd.read_csv('https://example.com/data.csv')
```

3.open() method : read a **CSV file** without using any external modules and store the data in a dictionary

```
# Open the CSV file
with open('file.csv', 'r') as f:
    lines = f.readlines()  # Read all lines into a list

# Split the first line to get the header
header = lines[0].strip().split(',')

# Initialize an empty list to hold rows as dictionaries
data = []

# Iterate over the remaining lines and map them to the header
for line in lines[1:]:
    values = line.strip().split(',')  # Split the line into values
    row = dict(zip(header, values))  # Map header to values
    data.append(row)
```

Other Methods :

```
import csv

# Open the file
with open('file.csv', 'r') as f:
    reader = csv.DictReader(f)  # Creates a reader object with column headers as keys
    data = [row for row in reader]  # Convert to a list of dictionaries
```

1.5 DATASETS

1.5.1 CSM (Conventional and Social Media Movies) Dataset 2014 and 2015.csv
1.5.2 Facebook Metrics.csv
1.5.3 Power Consumption of Tetouan City.csv
1.5.4 Steel Industry Energy Consumption.csv
1.5.5 Internet Firewall.csv

1.5.1 CSM (Conventional and Social Media Movies) Dataset 2014 and 2015.csv

- **Dataset Overview:**

1. Rows : 217
2. Columns : 12

- **Description:**

The dataset comprises information about smartphones, capturing various attributes such as brands, colors, memory, storage, user ratings, selling prices, original prices, and discounts. It offers a comprehensive view of the smartphone market, enabling analysis of pricing strategies, consumer preferences, and market trends. With detailed specifications and pricing data, the dataset serves as a valuable resource for retailers, manufacturers, and analysts seeking insights into the competitive landscape and consumer behavior within the smartphone industry.

The dataset will include the following attributes

Features	Types	Description
Brands	Categorical	The brands of smartphones
Models	Categorical	The Models of smartphones
Colors	Categorical	The colors of the smartphones.
Memory	Categorical	The storage capacity of the smartphones measured in gigabytes(GB)
Storage	Categorical	The internal storage capacity of the smartphones measured in gigabytes (GB)
Camera	Categorical	The camera of the smartphones described in megapixel(MP)
Rating	Numerical	The user ratings assigned to the smartphones
Selling Price	Numerical	The price at which the smartphones are sold to consumers
Original Price	Numerical	The original price of the smartphones before any discounts
Mobile	Categorical	Indicates whether the device is a mobile phone
Discount	Numerical	The discount applied to the original price to calculate the selling price.
DiscounPercentage	Numerical	The percentage discount applied to the original price to calculate the selling price.

1.5.2 Facebook Metrics.csv

- **Overview :**

Rows : 500
Columns : 18

- **Description:**

The data is related to posts' published during the year of 2014 on the Facebook's page of a renowned cosmetics brand.

This dataset contains 500 of the 790 rows and part of the features analyzed by Moro et al. (2016). The remaining were omitted due to confidentiality issues.

The dataset will include the following attributes

Features	Role	Type
Page total likes	Feature	Integer
Type	Feature	Categorical
Category	Feature	Integer
Post Month	Feature	Integer
Post Weekday	Feature	Integer
Post Hour	Feature	Integer
Paid	Feature	Continuous
Lifetime Post Total Reach	Feature	Integer
Lifetime Post Total Impressions	Feature	Integer
Lifetime Engaged Users	Feature	Integer
Lifetime Post Consumers	Feature	Integer
Lifetime Post Consumptions	Feature	Integer
Lifetime Post Impressions by people who have liked your Page	Feature	Integer
Lifetime Post reach by people who like your Page	Feature	Integer
Lifetime People who have liked your Page and engaged with your post	Feature	Integer
Comment	Feature	Integer
Like	Feature	Integer
Share	Feature	Integer
Total Interactions	Target	Integer

1.5.3 Power Consumption of Tetouan City.csv

- **Overview :**

1. Rows : 52417
2. Columns : 9

- **Description :**

This dataset captures power consumption data and weather measurements in Tetuan City, providing insights into the relationship between environmental factors and energy usage. It includes records of temperature, humidity, and wind speed alongside solar radiation measurements, such as general and diffuse flows. Additionally, it details the power consumption for three distinct zones in the city. The data, timestamped with precise date and time, allows for the analysis of daily and seasonal trends in weather and energy demand, facilitating a deeper understanding of patterns and potential correlations between climate conditions and energy usage..

The dataset will include the following attributes

Features	**Types**	**Description**
DateTime	Timeseries	Date and time of measurement.
Temperature	Quantitative	Recorded temperature in degrees Celsius (°C).
Humidity	Quantitative	Recorded humidity level as a percentage (%).
Wind Speed	Quantitative	Speed of the wind in meters per second (m/s).
General Diffuse Flows	Quantitative	General diffuse solar radiation in Watt per square meter (W/m²).
Diffuse Flows	Quantitative	Diffuse radiation measurement in Watt per square meter (W/m²).
Zone 1 Power Consumption	Quantitative	Power consumption in Zone 1 in kilowatt-hours (kWh).
Zone 2 Power Consumption	Quantitative	Power consumption in Zone 2 in kilowatt-hours (kWh).
Zone 3 Power Consumption	Quantitative	Power consumption in Zone 3 in kilowatt-hours (kWh).
DateTime	Timeseries	Date and time of measurement.

1.5.4 Steel Industry Energy Consumption.csv

- **Overview :**

1. Rows : 11251

2. Columns : 13

- **Description :**

The information gathered is from the DAEWOO Steel Co. Ltd in Gwangyang, South Korea. It produces several types of coils, steel plates, and iron plates. The information on electricity consumption is held in a cloud-based system. The information on energy consumption of the industry is stored on the website of the Korea Electric Power Corporation (pccs.kepco.go.kr), and the perspectives on daily, monthly, and annual data are calculated and shown.

Features	Types	Description
date	Date	The specific date on which the data was recorded.
Usage_kWh	Continuous	The total energy consumption measured in kilowatt-hours (kWh) for the given date.
Lagging_Current_Reactive.Power_kVarh	Continuous	The reactive power in kilovolt-ampere reactive hours (kVarh) consumed when the current lags the voltage.
Leading_Current_Reactive_Power_kVarh	Continuous	The reactive power in kilovolt-ampere reactive hours (kVarh) consumed when the current leads the voltage.
CO2(tCO2)	Continuous	The total carbon dioxide emissions measured in metric tons of CO2 (tCO2).
Lagging_Current_Power_Factor	Continuous	A measure of the efficiency of power usage when the current lags the voltage
Leading_Current_Power_Factor	Continuous	A measure of the efficiency of power usage when the current leads the voltage
NSM	Integer	A numerical representation of the number of seconds since midnight (NSM) for a given time.
WeekStatus	Categorical	Indicates whether the day falls on a weekday or a weekend.
Day_of_week	Categorical	The specific day of the week (e.g., Monday, Tuesday, etc.).
date	Date	typically formatted as YYYY-MM-DD
Usage_kWh	Continuous	The total energy consumed during the recorded time period, measured in kilowatt-hours (kWh).
Lagging_Current_Reactive.Power_kVarh	Continuous	measured in kilovolt-ampere reactive hours (kVarh)

1.5.5 Internet firewall.csv

- **Overview :**

1. Rows : 65532
2. Columns : 12

- **Description :**

This dataset captures information about internet traffic flows, providing details about network connections. It includes data about the ports used for communication, the amount of data sent and received, and the number of packets exchanged during the connection. Additionally, it records the duration of each connection and whether the action was allowed or blocked. This data is useful for analyzing network traffic patterns, troubleshooting issues, or studying firewall activity.

The dataset will include the following attributes

Features	**Types**	**Description**
Source Port	Numerical	Port number of the source device.
Destination Port	Numerical	Port number of the destination device.
NAT Source Port	Numerical	Port number translated by NAT for the source.
NAT Destination Port	Numerical	Port number translated by NAT for the destination.
Action	Categorical	Action taken (e.g., allow or block).
Bytes	Numerical	Total bytes transferred during the connection.
Bytes Sent	Numerical	Bytes sent from source to destination.
Bytes Received	Numerical	Bytes received from destination to source.
Packets	Numerical	Number of packets transmitted during the connection.
Elapsed Time (sec)	Numerical	Elapsed time of the connection in seconds.
pkts_sent	Numerical	Number of packets sent from the source.
pkts_received	Numerical	Number of packets received by the destination.
Source Port	Numerical	Port number of the source device.
Destination Port	Numerical	Port number of the destination device.

MODULE 2

DATA PROCESSING

2.1 Data Processing using Numpy and Pandas

Data processing refers to the systematic process of collecting, organizing, transforming, and analyzing raw data to extract meaningful insights and information. It involves a series of steps that clean and prepare data, making it ready for decision-making, reporting, or advanced analysis. This process is essential in converting messy, unstructured, or incomplete datasets into structured formats that can be easily understood and utilized. In the modern digital era, data processing is crucial across industries for tasks such as identifying trends, improving operations, and supporting strategic decisions. By leveraging tools like NumPy and Pandas, organizations can automate data processing workflows, handle large datasets efficiently, and maintain data integrity throughout the pipeline.

Packages Used :

Numpy : NumPy (Numerical Python) is a high-performance library for numerical computations, offering support for multi-dimensional arrays, matrices, and a range of mathematical operations. Its C-based implementation makes it ideal for tasks like linear algebra, statistics, and simulations.

Pandas : Pandas is a data analysis library built on NumPy, providing tools for working with structured data. Its key structures, Series and DataFrame, simplify tasks like data cleaning, grouping, and reshaping, making data manipulation intuitive and efficient.

Functions Used :

pd.read_csv() - Reads a CSV file into a Pandas DataFrame.
pd.to_csv() - Writes a DataFrame to a CSV file.
pd.DataFrame() - Creates a DataFrame from arrays, dictionaries, or other structures.
pd.Series() - Creates a one-dimensional labeled array, similar to a column in a spreadsheet.
df.head() - Displays the first n rows of a DataFrame (default is 5).
df.describe() - Generates summary statistics for numerical columns.
df.groupby() - Groups the DataFrame rows by specified column(s) for aggregation.
df.isnull() - Identifies missing (NaN) values in the DataFrame.

np.mean() - Calculates the arithmetic mean of array elements along a specified axis.

np.median() - Computes the median value of array elements.

np.std() - Calculates the standard deviation of array elements.

np.sum() - Returns the sum of array elements along a given axis.

np.max() - Finds the maximum value in an array.

np.min() - Finds the minimum value in an array.

np.arange() - Creates a one-dimensional array with evenly spaced values within a given range.

np.linspace() - Generates an array of evenly spaced numbers over a specified interval.

np.reshape() - Reshapes an array without changing its data.

np.concatenate() - Joins multiple arrays along a specified axis.

MODULE 3
DATA VISUALIZATION

Description

Data visualization is the graphical representation of information and data using visual elements like charts, graphs, and maps. It provides a way to communicate complex datasets in an intuitive, visually appealing manner, enabling easier comprehension and analysis. By leveraging patterns, trends, and outliers in data, visualization helps decision-makers derive actionable insights quickly and effectively. In today's data-driven world, visualization plays a pivotal role in understanding large datasets and communicating findings across various domains, such as business analytics, scientific research, and social media analysis. Tools like **Matplotlib** and **Seaborn** in Python are popular for creating both basic and advanced visualizations.

Importance of Data Visualization:

1. **Simplifies Complexity**: Converts raw data into clear and meaningful visuals.
2. **Identifies Trends**: Helps uncover relationships and patterns in data.
3. **Facilitates Decision-Making**: Enhances understanding for stakeholders and decision-makers.
4. **Engages Audience**: Visuals are more engaging and easier to interpret than raw numbers.
5. **Supports Big Data Analysis**: Essential for summarizing and exploring large datasets effectively

Libraries Used:

NumPy: NumPy (Numerical Python) is a high-performance library for numerical computations, offering support for multi-dimensional arrays, matrices, and a range of mathematical operations. Its C-based implementation makes it ideal for tasks like linear algebra, statistics, and simulations.

Pandas : Pandas is a data analysis library built on NumPy, providing tools for working with structured data. Its key structures, Series and DataFrame, simplify tasks like data cleaning, grouping, and reshaping, making data manipulation intuitive and efficient.

Functions Used :

Matplotlib Functions:

plt.plot() - Creates a line plot for trends and continuous data visualization.

plt.bar() - Plots a bar chart for comparing categorical data.

plt.scatter() - Generates a scatter plot for analyzing relationships between variables.

plt.hist() - Creates a histogram to visualize the frequency distribution of data.

plt.pie() - Draws a pie chart to represent proportions of categories.

plt.contour() - Draws contour lines for three-dimensional data projections.

plt.contourf() - Creates filled contour plots for enhanced 3D data interpretation.

plt.title() - Adds a title to the plot.

plt.xlabel() - Adds a label to the x-axis of the plot.

plt.ylabel() - Adds a label to the y-axis of the plot.

plt.legend() - Displays a legend to identify multiple datasets in the plot.

Seaborn Functions:

sns.barplot() - Creates a bar plot with aggregate values across a categorical variable.

sns.scatterplot() - Visualizes the relationship between two numeric variables.

sns.heatmap() - Displays a heatmap to show data correlations or intensity patterns.

sns.boxplot() - Summarizes the distribution of data using a box-and-whisker plot.

sns.violinplot() - Combines boxplot and KDE to show data distribution and spread.

sns.pairplot() - Creates scatter plots for all pairs of variables in a dataset.

sns.lineplot() - Plots a line graph for continuous data over a range or time.

sns.polarplot() - Seaborn does not directly support polar plots but can pass data to Matplotlib’s polar plot functionalities.

sns.kdeplot() - Plots a Kernel Density Estimate (KDE) to show the probability density of a variable.

sns.clustermap(data) - Generates a heatmap with hierarchical clustering of rows and columns.

MODULE - 4

DATA CLEANING AND POST VISUALIZATION

Description :

Data cleaning is the essential step in the data analysis pipeline where inaccuracies, inconsistencies, duplicates, and missing values in raw data are addressed to improve its quality and usability. This process ensures that the dataset is reliable and free from issues that could skew analysis results. Common cleaning tasks include removing duplicates, imputing missing values, correcting data types, and handling outliers. High-quality data leads to more accurate models and better decision-making.

Post-visualization comes after the cleaning phase, where the processed data is visualized to validate its integrity, detect remaining anomalies, and identify patterns. Visualizations like scatter plots, histograms, and box plots are often used to confirm the success of data cleaning and explore relationships within the dataset. This step ensures that the final dataset is ready for deeper analysis or modeling.

Libraries Used:

1. **Pandas**: Handling missing values, filtering data, and performing transformations.
2. **Scikit-learn**: Providing tools for imputing values, scaling, encoding, and normalization.
3. **Matplotlib and Seaborn**: Used for post-visualization to validate and explore data.

Functions Used :

Pandas Functions:

df.dropna() - Removes rows or columns with missing values.

df.fillna() - Fills missing values with specified values or methods like mean or median.

df.duplicated() - Identifies duplicate rows in a DataFrame.

df.drop_duplicates() - Removes duplicate rows from a DataFrame.

df.replace() - Replaces specific values in a DataFrame.

df.rename() - Renames columns or indexes.

df.isnull() - Checks for missing (NaN) values in the DataFrame.

df.interpolate() - Performs interpolation to fill missing values.

df.astype() - Converts data types of columns in a DataFrame.

df.query() - Filters data based on query conditions.

Scikit-learn Functions:

SimpleImputer() - Replaces missing values with statistical measures like mean, median, or mode.

KNNImputer() - Imputes missing values based on nearest neighbors.

MinMaxScaler() - Normalizes data to a range between 0 and 1.

StandardScaler() - Standardizes data to have zero mean and unit variance.

RobustScaler() - Scales data while being robust to outliers.

LabelEncoder() - Encodes categorical variables as numeric labels.

OneHotEncoder() - Converts categorical variables into one-hot encoded vectors.

normalize() - Normalizes arrays to have unit norm.

Pipeline() - Chains multiple preprocessing steps into a single workflow.

ColumnTransformer() - Applies different preprocessing steps to specific columns.

MODULE - 5

ADVANCED VISUALIZATION

Description :

Advanced visualization encompasses the creation of complex, layered, and often interactive representations of data to reveal intricate patterns and insights. Tools like **Basemap**, an extension of Matplotlib, specialize in geospatial data visualization by enabling users to create detailed maps and overlay data on geographic coordinates. With Basemap, tasks like plotting locations, tracing movement, analyzing climate patterns, or studying population density become straightforward and visually intuitive. These visualizations enhance comprehension of spatial relationships, making them essential for industries like environmental science, urban development, and disaster management. Advanced visualization transforms raw data into meaningful visuals, offering clarity and depth for data-driven decision-making.

Libraries Used:

Basemap is a toolkit built on top of Matplotlib, designed for geospatial data visualization. It allows users to create maps with various projections, add geographic features (e.g., coastlines, rivers), and overlay data on these maps. Basemap is widely used for plotting spatial relationships and analyzing geographic trends in datasets.

Functions Used :

- **Basemap()** - Initializes a map with specified projection and geographic boundaries.
- **drawcoastlines()** - Draws the coastlines on the map.
- **drawcountries()** - Plots country borders on the map.
- **drawstates()** - Adds state or province boundaries (if available).
- **drawrivers()** - Displays rivers on the map.
- **fillcontinents()** - Fills land areas with specified colors.
- **drawmapboundary()** - Draws the map boundary and optionally fills the background.
- **plot()** - Plots points or paths on the map using latitude and longitude coordinates.
- **scatter()** - Creates a scatter plot for geographical data points.

- **drawparallels()** - Adds latitude lines to the map.
- **drawmeridians()** - Adds longitude lines to the map.
- **imshow()** - Displays image data over the map (e.g., heatmaps).
- **bluemarble()** - Renders a NASA "Blue Marble" image as the map's background.
- **shadedrelief()** - Adds a shaded relief image to the map for terrain visualization.
- **etopo()** - Displays Earth topography data as a background.

MODULE - 6

Description :

Data Wrangling

Data wrangling is the process of cleaning, reshaping, and enriching raw data into a format suitable for analysis. This process involves handling missing data, fixing inconsistencies, transforming data types, and merging multiple datasets. Wrangling ensures that the dataset is accurate, complete, and properly structured, which is critical for reliable analysis and modeling. It is particularly important when working with real-world data, as it is often messy and unorganized. Effective data wrangling leads to higher-quality insights and reduces errors during analysis.

Data Transformation

Data transformation focuses on altering the structure, values, or features of data to fit the requirements of analysis or modeling. This includes techniques like normalization, scaling, encoding categorical variables, and creating new features through aggregation or combinations. Transformation often improves data interpretability, ensures compatibility with algorithms, and highlights underlying patterns. For instance, scaling numerical features is essential for machine learning models like gradient descent, which are sensitive to differences in feature magnitudes.

Importance of Wrangling and Transformation

1. **Data Consistency**: Ensures all values are in a standard format, reducing discrepancies.
2. **Improved Analysis**: Enables the extraction of accurate insights from clean and transformed data.
3. **Model Compatibility**: Prepares datasets to meet the specific requirements of analytical tools or algorithms.
4. **Error Prevention**: Reduces biases and errors caused by incomplete or inconsistent data.

Data Wrangling

Data wrangling, also known as data munging, is the process of preparing messy, unstructured, or incomplete data for analysis. It involves systematically identifying and addressing inconsistencies, outliers, and missing values to create a dataset that is clean and reliable. Wrangling often includes combining data from multiple sources, reformatting data for consistency, and ensuring the dataset is free from redundancies or inaccuracies.

- **Handling Missing Values**: Missing data can skew results and must be addressed through imputation (filling gaps with statistical values) or deletion (removing rows/columns with too many nulls).
- **Outlier Detection and Removal**: Identifying extreme values that may distort analysis and either removing or capping them for consistency.
- **Reshaping Data**: Converting between wide and long formats using tools like pivot() or melt() to fit analysis needs.
- **Combining Datasets**: Merging, concatenating, or joining multiple datasets for richer analysis

Data Transformation

Data transformation is a more technical aspect of preparing data for specific types of analysis or machine learning models. It focuses on converting raw features into representations that are more suitable for downstream tasks.

- **Scaling**: Adjusting numerical values to a standard range or distribution using techniques like min-max scaling or standardization.
- **Encoding Categorical Variables**: Converting text-based categorical data into numeric formats using methods like one-hot encoding or label encoding.
- **Feature Engineering**: Creating new features or transforming existing ones to highlight relationships or improve model performance. For instance, combining "date" columns into "day of the week" or "month" for trend analysis.
- **Aggregation**: Summarizing data by grouping it into meaningful categories (e.g., total sales by region).
- **Normalization**: Ensuring numerical features have similar magnitudes to avoid dominance in models sensitive to scale.

Challenges in Wrangling and Transformation

1. **Complexity of Data**: Diverse data formats and inconsistent sources make integration challenging.

2. **High Dimensionality**: Datasets with many features may require careful selection and transformation to focus on relevant data.
3. **Automation**: While wrangling and transformation can be automated, nuanced cleaning tasks often require manual intervention.
4. **Time-Intensive**: These steps can consume significant time, especially for large datasets.

MODULE - 7

Description :

R : R is a programming language and environment specifically designed for statistical computing and data analysis. It provides a comprehensive suite of tools for data manipulation, statistical modeling, and visualization. R is highly favored by statisticians and data scientists because of its wide range of libraries for handling complex data and performing detailed analyses. One of its core strengths lies in its visualization capabilities, with packages like ggplot2 enabling the creation of powerful, customizable plots and charts. R is ideal for working with large datasets, conducting complex statistical tests, and producing publication-quality graphics.

PowerBI : Power BI is a business analytics tool developed by Microsoft that enables users to visualize and share insights from their data. Power BI provides a range of powerful tools for transforming raw data into interactive reports, dashboards, and visualizations. It integrates seamlessly with various data sources, including databases, Excel files, and cloud services, making it a flexible tool for both business and technical users. Power BI is widely used in organizations to track KPIs, monitor business trends, and perform data analysis in an interactive way.

Building Interactive Dashboards with Power BI

Power BI stands out in the realm of data visualization for its user-friendly interface, which allows even non-technical users to create sophisticated visualizations without extensive programming knowledge. With its drag-and-drop functionality, users can build interactive dashboards that enable real-time data exploration and decision-making.

Some key features of Power BI that make it a powerful tool for building dashboards and visualizations include:

1. **Data Connectivity**: Power BI can connect to a wide variety of data sources, including databases (SQL Server, MySQL), cloud services (Azure, Google Analytics), and even flat files like Excel.

2. **Interactive Visualizations**: Power BI allows users to create dynamic reports with clickable filters, drill-down capabilities, and the ability to interact with charts, maps, and tables.
3. **Customizable Dashboards**: Dashboards in Power BI can be tailored to meet the needs of different users, enabling stakeholders to view KPIs, trends, and insights in real-time.
4. **Integration with Excel and other Microsoft Products**: Power BI works seamlessly with Excel, SharePoint, and other Microsoft tools, making it easy to import and manipulate data.
5. **Real-time Data**: With Power BI's real-time streaming, users can track live data, such as sales figures, web traffic, or inventory levels, to make on-the-spot decisions.
6. **Advanced Analytics**: Power BI supports advanced analytics features like forecasting, trend analysis, and the use of DAX (Data Analysis Expressions) to create complex calculations.
7. **Collaboration**: Dashboards and reports in Power BI can be shared with others through the cloud, facilitating team collaboration and data-driven decision-making.

PowerBI Desktop:

Power BI Desktop is a free application provided by Microsoft that allows users to create reports and data visualizations on their local computers. It is a powerful tool for data analysts and business intelligence professionals to connect to various data sources, clean and transform data, and build interactive visualizations. Once reports are created in Power BI Desktop, they can be published to the Power BI Service (cloud) for sharing and collaboration.

Power BI Desktop combines the capabilities of data transformation, data modeling, and visualization in one tool, making it an essential part of the Power BI suite. It is mainly used for designing, creating, and authoring reports, which can then be distributed across the organization via the Power BI Service.

Key Features of Power BI Desktop

Data Connectivity:
Power BI Desktop can connect to a wide variety of data sources, including databases (SQL Server, Oracle, MySQL) , web data , flat files (Excel , CSV), cloud-based services (Azure, Google Analytics), and many others. It also supports direct query mode, where data remains in the source and queries are run in real-time.

Data Transformation with Power Query:
The Power Query Editor allows users to clean , transform , and prepare data for analysis. Users can filter

rows, merge tables, pivot data, and perform other data wrangling tasks, all with an intuitive drag-and-drop interface. This feature makes it easy to shape raw data before loading it into Power BI.

Data Modeling:
Power BI Desktop provides a robust data modeling interface where users can create relationships between tables, build calculated columns, measures, and key performance indicators (KPIs) using **DAX (Data** Analysis Expressions**)**. These models can define how different tables are linked and enable complex calculations on the data.

Custom Visualizations:
Power BI Desktop allows users to create custom visuals to represent data in unique ways. In addition to the built-in visuals like bar charts, pie charts, and maps, users also import custom visuals from the Power BI marketplace or develop their own.

Report Authoring:
Power BI Desktop provides a canvas where users can place various visualizations like charts, tables, and maps to build reports. Users can also apply themes, design custom layouts, and make their reports interactive by adding slicers, filters, and drill-through actions.

Advanced Analytics:
Power BI Desktop includes advanced analytical capabilities, such as statistical modeling, trend analysis, and forecasting. It also supports integration with Python and R scripts for more complex calculations and visualizations.

Drag-and-Drop Interface:
With Power BI Desktop's intuitive drag-and-drop interface, users can easily move data fields onto the report canvas to create visuals. This simplicity makes Power BI accessible even for users who are not experienced in coding or data visualization.

CONCLUSION

This mini project focused on analyzing, visualizing, and transforming datasets using various tools and techniques explored across Modules 1 to 7. Key steps included data acquisition, preprocessing, cleaning, visualization, and advanced data transformation. Python libraries like NumPy, Pandas, Matplotlib, and Seaborn were utilized for efficient data handling and analysis, enabling valuable insights and trends to be extracted from raw data. Additionally, integrating R and Power BI expanded the analytical scope, offering advanced statistical modeling, visualization, and interactive dashboard-building capabilities.

Data wrangling and transformation were central to preparing the data for meaningful analysis. Power BI’s Power Query Editor facilitated efficient cleaning and structuring, addressing missing values and inconsistencies. Techniques such as normalization and scaling using Scikit-learn ensured the data was standardized for deeper analysis. Power BI's interactive dashboards enabled real-time data visualization, crucial for decision-making and stakeholder communication.

Visualization played a pivotal role in exploring trends, correlations, and patterns. Using Seaborn and Power BI, we created visual tools such as heatmaps, bar charts, and interactive dashboards, making data insights more accessible and actionable. R further supported statistical modeling and advanced visualizations, enriching the dataset's analysis and understanding.

This mini project underscores the importance of data cleaning, transformation, and visualization in modern analytics. By integrating Python, R, and Power BI, large datasets were processed efficiently, hidden patterns uncovered, and insights presented in user-friendly formats. Adopting these best practices empowers organizations to make data-driven decisions and gain a competitive edge.

REFERENCES

[1] **McKinney, W. (2017)**. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. O'Reilly Media.

[2] **Waskom, M., et al. (2020)**. *Seaborn: Statistical Data Visualization*. Journal of Open Source Software, 5(51), 2187.

[3] **Hunter, J. D. (2007)**. *Matplotlib: A 2D Graphics Environment*. Computing in Science & Engineering, 9(3), 90-95.

[4] **Microsoft. (2020)**. *Power BI Desktop Overview*. Retrieved from https://docs.microsoft.com/en-us/power-bi/fundamentals/desktop-getting-started

[5] **Wickham**, **H**., et al. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag.

[6] **VanderPlas, J. (2016).** *Python Data Science Handbook: Essential Tools for Working with Data.* O'Reilly Media.
(Focuses on Python libraries such as Pandas, NumPy, Matplotlib, Scikit-learn, and Jupyter.)

[7] **Harris, C. R., et al. (2020).** *Array programming with NumPy. Nature*, 585(7825), 357-362.
(Explores the capabilities and usage of the NumPy library in scientific computing.)

[8] **Pedregosa, F., et al. (2011).** *Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research*, 12, 2825-2830.
(Discusses machine learning techniques, including data preprocessing and scaling.)

[9] **Wickham, H., & Grolemund, G. (2017).** *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data.* O'Reilly Media.
(A comprehensive guide to using R for data manipulation and visualization.)

[10] **McKinney, W., & Perktold, J. (2011).** *Data Structures for Statistical Computing in Python.Proceedings of the 10th Python in Science Conference (SciPy).*
(Explains how Pandas and NumPy streamline statistical analysis and data manipulation.)