**ASSIGNMENT: - 02**

# **Problem Statement: -**

Perform the following operations using R/Python on the data sets:

a) Compute and display summary statistics for each feature available in the dataset. (e.g. minimum value, maximum value, mean, range, standard deviation, variance and percentiles

b) Data Visualization-Create a histogram for each feature in the dataset to illustrate the feature distributions.

c) Data cleaning, Data integration, Data transformation, Data model building (e.g. Classification)

# **S/W, Library, Package: -**

1. Software: Python
   * Python is a high-level programming language widely used in data science, machine learning, and scientific computing.
2. Library: pandas
   * pandas are a powerful data manipulation and analysis library for Python.
   * It provides data structures like DataFrames and Series, along with functions for reading/writing data, indexing, selecting, sorting, and handling missing values.
3. Packages:

For reading data from different formats:

* + CSV: pandas can directly read CSV files using pd. read\_csv ().
  + Excel: openpyxl or xlrd packages might be required alongside pandas for reading Excel files (pd. read\_excel ()).

For numerical operations and data manipulation:

* + pandas: Used extensively for indexing, selecting, sorting, describing data attributes, counting unique values, converting data types, and handling missing values.

NumPy: Often used in conjunction with pandas for numerical operations and statistical calculations.

* For formatting and applying custom functions:

Python's built-in functions (lambda functions) are used along with pandas apply () method for custom formatting.

* For statistical calculations and imputing missing values:

SciPy: Can be used for more advanced statistical functions if needed.

# **Theory:**

a) **Compute and display summary statistics**:

**Methodology:**

* In R, you can use the **summary ()** function or specific functions like **min ()**, **max ()**, **mean ()**, **range ()**, **sd ()** (standard deviation), **var ()** (variance), and **quantile ()** to compute summary statistics.
* In Python, pandas provide the **describe ()** method to generate summary statistics like count, mean, standard deviation, minimum, 25th percentile, median (50th percentile), 75th percentile, and maximum.

**Advantages:**

* Provides a quick overview of the dataset's numerical attributes.
* Helps in understanding the central tendency, spread, and distribution of data.

**Applications**:

* Data exploration and initial analysis.
* Identifying outliers or anomalies in data.

**Limitations**:

* Summary statistics may not capture the full picture of complex data distributions.
* Sensitivity to extreme values (outliers) can skew results.

**Example in Python**:

import pandas as pd

# Assuming 'df' is your dataframe

Summary\_stats = df.describe()

print(summary\_stats)

b) **Data Visualization - Create histograms**:

**Methodology**:

* In R, use the **hist()** function to create histograms for each feature.
* In Python, use libraries like matplotlib or seaborn to plot histograms.

**Advantages**:

* Visualizes the distribution of data for each feature.
* Helps in identifying patterns, skewness, and outliers.

**Applications**:

* Exploratory data analysis (EDA).
* Presenting data distributions in reports or presentations.

**Limitations**:

* Histograms may not capture complex data relationships.
* Interpretation can be subjective based on bin sizes and visualization parameters.

**Example in R**:

# Assuming 'df' is your dataframe and 'feature' is the feature column hist(df$feature, main="Histogram of Feature")

c) **Data Cleaning, Data Integration, Data Transformation, Data Model Building**:

**Methodology**:

* Data cleaning involves handling missing values, outliers, and inconsistencies.
* Data integration merges datasets if needed, combining data from different sources.
* Data transformation includes feature scaling, encoding categorical variables, and creating new features.
* Data model building involves selecting a suitable algorithm (e.g., classification, regression) and training the model on the cleaned and transformed data.

**Advantages**:

* Improves data quality and consistency.
* Enhances data compatibility for analysis and modeling.
* Prepares data for machine learning algorithms.

**Applications**:

* Preprocessing data for statistical analysis or machine learning.
* Building predictive models for classification or regression tasks.

**Limitations**:

* Data cleaning and transformation processes can be time-consuming.
* Data integration may introduce inconsistencies or errors if not done carefully.

# **Working/ Algorithm:**

a) Compute and display summary statistics for each feature available in the dataset. (e.g. minimum value, maximum value, mean, range, standard deviation, variance and percentiles

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# import the data

data= pd.read\_csv("/content/insurance.csv")

data.head()

data.describe()

data.var()

b) Data Visualization-Create a histogram for each feature in the dataset to illustrate the feature distributions.

a=data["age"]

plt.hist(a,edgecolor="black")

plt.xlabel("age")

plt.title("age")

plt.legend()

plt.show

b=data["bmi"]

plt.hist(b,color="pink",edgecolor="black")

plt.title("bmi")

plt.show

c=data["children"]

plt.hist(c,alpha=0.5,color="green",edgecolor="black")

plt.title("children")

plt.show

d=data["charges"]

plt.hist(d,color="yellow",edgecolor="black")

plt.title("charges")

plt.show

c) Data cleaning, Data integration, Data transformation, Data model building (e.g. Classification)

# **Conclusion:**

Both R and Python provide powerful tools for data analysis and manipulation. Summary statistics and histograms help understand data distributions and characteristics. Data cleaning, integration, transformation, and model building are crucial for preparing data for analysis and modeling tasks. Both languages offer versatile libraries and functionalities to cater to diverse data science needs.