

STES's
SINHGAD COLLEGE OF ENGINEERING
Vadgaon(Bk), Pune

Department of Computer Engineering



LABORATORY MANUAL

2023-24

**DATA SCIENCE AND BIG DATA
ANALYTICS LABORATORY**

TE-COMPUTER ENGINEERING

SEMESTER-II

Subject Code:310256

TEACHING SCHEME

Practical: 2 Hrs/Week

EXAMINATION SCHEME

Practical: 25 Marks

Term Work: 50 Marks

-: Name of Faculty:-

Prof. A.A.Kotalwar

Prof. A.M.Karanjkar

Prof. P.D.Bendale

Prof. S.S.Peerzade

Prof. B.R.Ban

Course Objectives:

- To understand principles of Data Science for the analysis of real time problems.
- To develop in depth understanding and implementation of the key technologies in Data Science and Big Data Analytics
- To analyze and demonstrate knowledge of statistical data analysis techniques for decision-making
- To gain practical, hands-on experience with statistics programming languages and Big Data tools

Course Outcomes: On completion of the course, learners will be able to

CO1: Apply principles of Data Science for the analysis of real time problems

CO2: Implement data representation using statistical methods

CO3: Implement and evaluate data analytics algorithms

CO4: Perform text preprocessing

CO5: Implement data visualization techniques

CO6: Use cutting edge tools and technologies to analyze Big Data.

INDEX

GROUP A: DATA SCIENCE

Sr.No.	Title	Page Number
1	Data Wrangling, I Perform the following operations using Python on any open source dataset (e.g., data.csv) <ol style="list-style-type: none"> 1. Import all the required Python Libraries. 2. Locate an open source data from the web (e.g. https://www.kaggle.com). Provide a clear description of the data and its source (i.e., URL of the web site). <ol style="list-style-type: none"> 3. Load the Dataset into pandas data frame. 4. Data Preprocessing: check for missing values in the data using pandas <code>isnull()</code>, <code>describe()</code> function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame. 5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions. 6. Turn categorical variables into quantitative variables in Python. In addition to the codes and outputs, explain every operation that you do in the above steps and explain everything that you do to import/read/scrape the data set.	7
2	Data Wrangling II Create an “Academic performance” dataset of students and perform the following operations using Python. <ol style="list-style-type: none"> 1. Scan all variables for missing values and inconsistencies. If there are missing 	23

	<p>values and/or inconsistencies, use any of the suitable techniques to deal with them.</p> <ol style="list-style-type: none"> 2. Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them. 3. Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable, to convert a non-linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution. Reason and document your approach properly. 	
3	<p>Descriptive Statistics - Measures of Central Tendency and variability</p> <p>Perform the following operations on any open source dataset (e.g., data.csv)</p> <ol style="list-style-type: none"> 1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable. 2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris- versicolor' of iris.csv dataset. Provide the codes with outputs and explain everything that you do in this step. 	34
4	<p>Data Analytics I</p> <p>Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset.</p> <p>The objective is to predict the value of prices of the house using the given features.</p>	39
5	<p>Data Analytics II</p> <ol style="list-style-type: none"> 1. Implement logistic regression using Python/R to perform classification on Social_Network_Ads.csv dataset. 2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset. 	48
6	<p>Data Analytics III</p> <ol style="list-style-type: none"> 1. Implement Simple Naïve Bayes classification algorithm using Python/R on iris.csv dataset. 2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset. 	53
7	<p>Text Analytics</p> <ol style="list-style-type: none"> 1. Extract Sample document and apply following document preprocessing 	58

	<p>methods:Tokenization, POS Tagging, stop words removal, Stemming and Lemmatization.</p> <p>2. Create representation of document by calculating Term Frequency and Inverse Document Frequency.</p>	
8	<p>Data Visualization I</p> <p>1. Use the inbuilt dataset 'titanic'. The dataset contains 891 rows and contains information about the passengers who boarded the unfortunate Titanic ship. Use the Seaborn library to see if we can find any patterns in the data.</p> <p>2. Write a code to check how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram.</p>	63
9	<p>Data Visualization II</p> <p>1. Use the inbuilt dataset 'titanic' as used in the above problem. Plot a box plot for distribution of age with respect to each gender along with the information about whether they survived or not. (Column names : 'sex' and 'age') Write observations on the inference from the above statistics.</p>	70
10	<p>Data Visualization III</p> <p>Download the Iris flower dataset or any other dataset into Data Frame.(e.g., https://archive.ics.uci.edu/ml/datasets/Iris). Scan the dataset and give the inference as:</p> <p>1. List down the features and their types (e.g., numeric, nominal) available in the dataset.</p> <p>2. Create a histogram for each feature in the dataset to illustrate the feature distributions.</p> <p>3. Create a box plot for each feature in the dataset. Compare distributions and identify outliers.</p>	75

GROUP B: BIG DATA ANALYTICS

Any 3 assignments are mandatory.

Sr.No.	Title	Page Number
1	Write a code in JAVA for a simple Word Count application that counts the number of occurrences of each word in a given input set using the Hadoop Map-Reduce framework on local-standalone set-up.	81
2	Design a distributed application using Map-Reduce which processes a log file of a system.	87
3	Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text input files and finds average for temperature, dew point and wind speed.	92
4	Write a simple program in SCALA using Apache Spark framework.	96

GROUP C: MINI PROJECT/CASE STUDY PYTHON/R

Any 2 mini project are mandatory.

Sr.No.	Title	Page Number
1	Write a case study on Global Innovation Network and Analysis (GINA). Components of analyticplan are 1. Discovery business problem framed, 2. Data, 3. Model planning analytic technique and 4. Results and Key findings.	103
2	Use the following dataset and classify tweets into positive and negative tweets. https://www.kaggle.com/ruchi798/data-science-tweets	104
3	Develop a movie recommendation model using the scikit-learn library in python. Refer dataset https://github.com/rashida048/Some-NLP-Projects/blob/master/movie_dataset.csv	105
4	Use the following covid_vaccine_statewise.csv dataset and perform following analytics on thegiven dataset https://www.kaggle.com/sudalairajkumar/covid19-inndia?select=covid_vaccine_statewise.csv a. Describe the dataset b. Number of persons state wise vaccinated for first dose in India c. Number of persons state wise vaccinated for second dose in India d. Number of Males vaccinated d. Number of females vaccinated	106
5	Write a case study to process data driven for Digital Marketing OR Health care systems withHadoop Ecosystem components as shown. (Mandatory) <ul style="list-style-type: none"> • HDFS: Hadoop Distributed File System • YARN: Yet Another Resource Negotiator • MapReduce: Programming based Data Processing • Spark: In-Memory data processing • PIG, HIVE: Query based processing of data services • HBase: NoSQL Database (Provides real-time reads and writes) • Mahout, Spark MLLib: (Provides analytical tools) Machine Learning algorithmlibraries Solar, Lucene: Searching and Indexing 	107

CO-PO Mapping Matrix

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO 1	1	1	1	2	2	2	-	-	-	-	3	-
CO 2	2	2	1	2	-	2	-	-	-	-	-	-
CO 3	1	1	1	2	-	1	-	-	-	-	-	-
CO 4	2	2	1	2	2	3	-	-	-	-	-	-
CO 5	2	1	2	3	2	-	-	-	-	-	-	-
CO 6	1	1	2	2	2	-	-	-	-	-	-	-

GROUP A: DATA SCIENCE

Assignment No.	1 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 1

Title: Perform the following operations using Python on any open source dataset (e.g., data.csv)
Import all the required Python Libraries.

1. Locate open source data from the web (e.g. <https://www.kaggle.com>).
2. Provide a clear description of the data and its source (i.e., URL of the web site).
3. Load the Dataset into the pandas data frame.
4. Data Preprocessing: check for missing values in the data using pandas `isnull()`, `describe()` function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.
5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.
6. Turn categorical variables into quantitative variables in Python.

Prerequisites: 1. Basic of Python Programming

2. Concept of Data Preprocessing, Data Formatting, Data Normalization and Data Cleaning.

Objectives: Students should be able to perform the data wrangling operation using Python on any open source dataset.

Theory:

1. Introduction to Dataset

A dataset is a collection of records, similar to a relational database table. Records are similar to table rows, but the columns can contain not only strings or numbers, but also nested data structures such as lists, maps, and other records.

Instance	x	y	z	class
	0.5351795492	0.9443102776	0.1582435145	1
	0.2372136153	0.6406416746	0.2375491506	1
	0.9115356348	0.3311024322	0.5615073269	0
	0.5634070287	0.4183148035	0.151904445	0
	0.3728975195	0.3816657621	0.616341473	1
	0.6783527289	0.938524515	0.5269012505	1
	0.09568660734	0.04465749689	0.0133451798	0
	0.2173318229	0.6170559076	0.3122273853	1
	0.818890594	0.7459451367	0.9026713492	0
	0.6064854042	0.5945985792	0.2188024961	0
	0.1546966824	0.1579937453	0.1333579164	0

Instance: A single row of data is called an instance. It is an observation from the domain.

Feature: A single column of data is called a feature. It is a component of an observation and is also called an attribute of a data instance. Some features may be inputs to a model (the predictors) and others may be outputs or the features to be predicted.

Data Type: Features have a data type. They may be real or integer-valued or may have a categorical or ordinal value. You can have strings, dates, times, and more complex types, but typically they are reduced to real or categorical values when working with traditional machine learning methods.

Datasets: A collection of instances is a dataset and when working with machine learning methods we typically need a few datasets for different purposes.

Training Dataset: A dataset that we feed into our machine learning algorithm to train our model.

Testing Dataset: A dataset that we use to validate the accuracy of our model but is not used to train the model. It may be called the validation dataset.

Data Represented in a Table:

Data should be arranged in a two-dimensional space made up of rows and columns. This type of data structure makes it easy to understand the data and pinpoint any problems. An example of some raw data stored as a CSV (comma separated values).

```
1., Avatar, 18-12-2009, 7.8
2., Titanic, 18-11-1997,
3., Avengers Infinity War, 27-04-2018, 8.5
```

The representation of the same data in a table is as follows:

S.No	Movie	Release Date	Ratings (IMDb)
1.	Avatar	18-12-2009	7.8
2.	Titanic	18-11-1997	Na
3.	Avengers Infinity War	27-04-2018	8.5

Pandas Data Types

A data type is essentially an internal construct that a programming language uses to understand how to store and manipulate data. A possible confusing point about panda's data types is that there is some overlap between pandas, python and numpy. This table summarizes the key points:

Pandas dtype	Python type	NumPy type	Usage
object	str or mixed	string_, unicode_, mixed types	Text or mixed numeric and non-numeric values
int64	int	int_, int8, int16, int32, int64, uint8, uint16, uint32, uint64	Integer numbers
float64	float	float_, float16, float32, float64	Floating point numbers
bool	bool	bool_	True/False values
datetime64	NA	datetime64[ns]	Date and time values
timedelta[ns]	NA	NA	Differences between two datetimes
category	NA	NA	Finite list of text values

2. Python Libraries for Data Science

a. Pandas: Pandas is an open-source Python package that provides high-performance, easy-to-use data structures and data analysis tools for the labeled data in Python programming language.

What can you do with Pandas?

1. Indexing, manipulating, renaming, sorting, merging data frame
2. Update, Add, Delete columns from a data frame
3. Impute missing files, handle missing data or NaNs
4. Plot data with histogram or box plot

b. NumPy

One of the most fundamental packages in Python, NumPy is a general-purpose array-processing package. It provides high-performance multidimensional array objects and tools to work with the arrays. NumPy is an efficient container of generic multi-dimensional data. NumPy's main object is the homogeneous multidimensional array. It is a table of elements or numbers of the same datatype, indexed by a tuple of positive integers. In NumPy, dimensions are called axes and the number of axes is called rank. NumPy's array class is called array aka array.

What can you do with NumPy?

1. Basic array operations: add, multiply, slice, flatten, reshape, index arrays
2. Advanced array operations: stack arrays, split into sections, broadcast arrays
3. Work with DateTime or Linear Algebra
4. Basic Slicing and Advanced Indexing in NumPy Python

c. Matplotlib

This is undoubtedly my favorite and a quintessential Python library. You can create stories with the data visualized with Matplotlib. Another library from the SciPy Stack, Matplotlib plots 2D figures.

What can you do with Matplotlib?

Histogram, bar plots, scatter plots, area plot to pie plot, Matplotlib can depict a wide range of visualizations. With a bit of effort and tint of visualization capabilities, with Matplotlib, you can create just any visualizations:

Line plots

- Scatter plots
- Area plots
- Bar charts and Histograms
- Pie charts
- Stem plots
- Contour plots
- Quiver plots
- Spectrograms

Matplotlib also facilitates labels, grids, legends, and some more formatting entities with Matplotlib.

d. Seaborn

So when you read the official documentation on Seaborn, it is defined as the data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. Putting it simply, seaborn is an extension of Matplotlib with advanced features.

e. 5. Scikit Learn

Introduced to the world as a Google Summer of Code project, Scikit Learn is a robust machine learning library for Python. It features ML algorithms like SVMs, random forests, k-means clustering, spectral clustering, mean shift,

cross-validation and more... Even NumPy, SciPy and related scientific operations are supported by Scikit Learn with Scikit Learn being a part of the SciPy Stack.

What can you do with Scikit Learn?

1. Classification: Spam detection, image recognition
2. Clustering: Drug response, Stock price
3. Regression: Customer segmentation, Grouping experiment outcomes
4. Dimensionality reduction: Visualization, Increased efficiency
5. Model selection: Improved accuracy via parameter tuning
6. Pre-processing: Preparing input data as a text for processing with machine learning algorithms.

3. Description of Dataset:

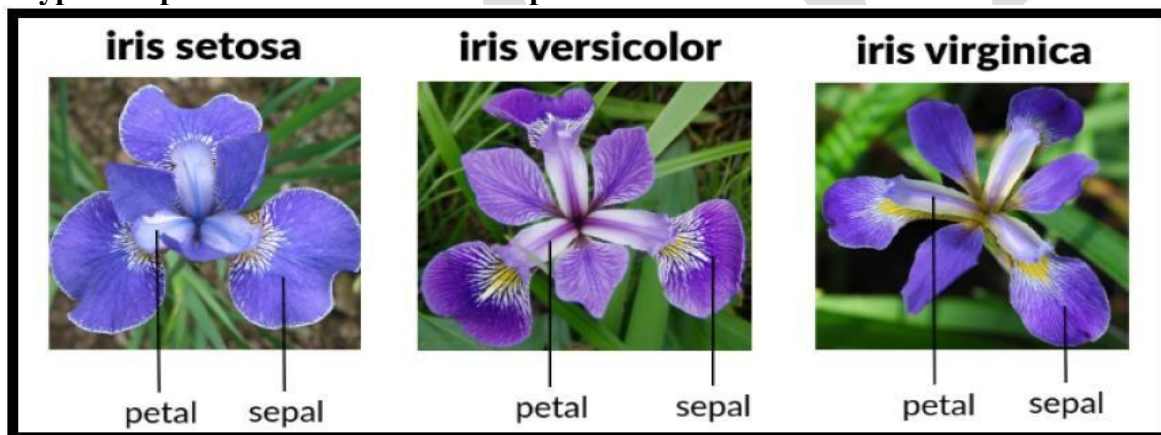
The Iris dataset was used in R.A. Fisher's classic 1936 paper, The Use of Multiple Measurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository. It includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other.

Total Sample- 150

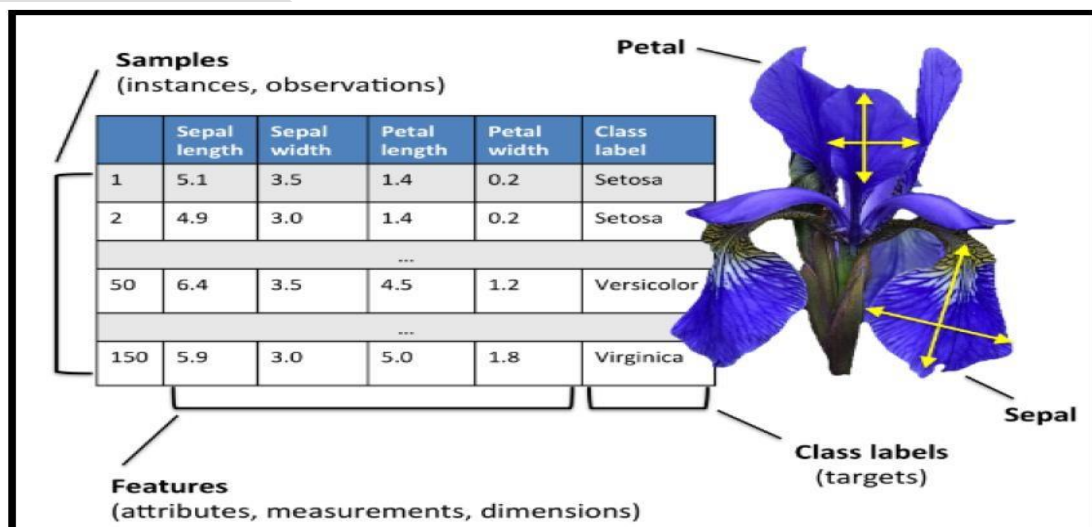
The columns in this dataset are:

1. Id
2. SepalLengthCm
3. SepalWidthCm
4. PetalLengthCm
5. PetalWidthCm
6. Species

3 Different Types of Species each contain 50 Sample-



Description of Dataset-



4. Panda Dataframe functions for Load Dataset

The columns of the resulting DataFrame have different dtypes.

iris.dtypes

1. The dataset is downloads from UCI repository.

csv_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

2. Now Read CSV File as a Dataframe in Python from from path where you saved the same The Iris data set is stored in .csv format. '.csv' stands for comma separated values. It is easier to load .csv files in Pandas data frame and perform various analytical operations on it.

Load Iris.csv into a Pandas data frame —

Syntax iris = pd.read_csv(csv_url, header = None)

3. The csv file at the UCI repository does not contain the variable/column names. They are located in a separate file.

col_names = ['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width', 'Species']

4. read in the dataset from the UCI Machine Learning Repository link and specify column names to use

iris = pd.read_csv(csv_url, names = col_names)

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

4. Panda Dataframe functions for Data Preprocessing: Dataframe Operations:

Sr. No	Data Frame Function	Description
1	<code>dataset.head(n=5)</code>	Return the first n rows.
2	<code>dataset.tail(n=5)</code>	Return the last n rows.
3	<code>dataset.index</code>	The index (row labels) of the Dataset.
4	<code>dataset.columns</code>	The column labels of the Dataset.
5	<code>dataset.shape</code>	Return a tuple representing the dimensionality of the Dataset.
6	<code>dataset.dtypes</code>	Return the dtypes in the Dataset. This returns a Series with the data type of each column. The result's index is the original Dataset's columns.

		Columns with mixed types are stored with the object dtype.
7	<code>dataset.columns.values</code>	Return the columns values in the Dataset in array format
8	<code>dataset.describe(include='all')</code>	Generate descriptive statistics. to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. Analyzes both numeric and object series, as well as Dataset column sets of mixed data types.
9	<code>dataset['Column name']</code>	Read the Data Column wise.
10	<code>dataset.sort_index(axis=1, ascending=False)</code>	Sort object by labels (along an axis).
11	<code>dataset.sort_values(by="Column name")</code>	Sort values by column name.
12	<code>dataset.iloc[5]</code>	Purely integer-location based indexing for selection by position.
13	<code>dataset[0:3]</code>	Selecting via [], which slices the rows.
14	<code>dataset.loc[:, ["Col_name1", "col_name2"]]</code>	Selection by label

Few Examples of iLoc to slice data for iris Dataset:

Sr. No	Data Frame Function	Description	Output																				
1	dataset.iloc[3:5, 0:2]	Slice the data	<table><tr><th>Id</th><th>Sepal LengthCm</th></tr><tr><td>3</td><td>4.6</td></tr><tr><td>4</td><td>5.0</td></tr></table>	Id	Sepal LengthCm	3	4.6	4	5.0														
Id	Sepal LengthCm																						
3	4.6																						
4	5.0																						
2	dataset.iloc[[1, 2, 4], [0, 2]]	By lists of integer position locations, similar to the NumPy/Python style:	<table><tr><th>Id</th><th>Sepal WidthCm</th></tr><tr><td>1</td><td>3.0</td></tr><tr><td>2</td><td>3.2</td></tr><tr><td>4</td><td>3.6</td></tr></table>	Id	Sepal WidthCm	1	3.0	2	3.2	4	3.6												
Id	Sepal WidthCm																						
1	3.0																						
2	3.2																						
4	3.6																						
3	dataset.iloc[1:3, :]	For slicing rows explicitly:	<table><tr><th>Id</th><th>SepalLengthCm</th><th>SepalWidthCm</th><th>PetalLengthCm</th><th>PetalWidthCm</th><th>Species</th></tr><tr><td>1</td><td>2</td><td>4.9</td><td>3.0</td><td>1.4</td><td>0.2</td><td>Iris-setosa</td></tr><tr><td>2</td><td>3</td><td>4.7</td><td>3.2</td><td>1.3</td><td>0.2</td><td>Iris-setosa</td></tr></table>	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	1	2	4.9	3.0	1.4	0.2	Iris-setosa	2	3	4.7	3.2	1.3	0.2	Iris-setosa
Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species																		
1	2	4.9	3.0	1.4	0.2	Iris-setosa																	
2	3	4.7	3.2	1.3	0.2	Iris-setosa																	
4	dataset.iloc[:, 1:3]	For slicing Column explicitly:	<table><tr><th>SepalLengthCm</th><th>SepalWidthCm</th></tr><tr><td>0</td><td>5.1</td><td>3.5</td></tr><tr><td>1</td><td>4.9</td><td>3.0</td></tr><tr><td>2</td><td>4.7</td><td>3.2</td></tr><tr><td>3</td><td>4.6</td><td>3.1</td></tr></table>	SepalLengthCm	SepalWidthCm	0	5.1	3.5	1	4.9	3.0	2	4.7	3.2	3	4.6	3.1						
SepalLengthCm	SepalWidthCm																						
0	5.1	3.5																					
1	4.9	3.0																					
2	4.7	3.2																					
3	4.6	3.1																					
4	dataset.iloc[1, 1]	For getting a value explicitly:	4.9																				
5	dataset['SepalLengthCm'].iloc[5]	Accessing Column and Rows by position	5.4																				

6	<code>cols_2_4=dataset.columns[2:4]</code> <code>dataset[cols_2_4]</code>	Get Column Name then get data from column	<table><tr><th></th><th>SepalWidthCm</th><th>PetalLengthCm</th></tr><tr><td>0</td><td>3.5</td><td>1.4</td></tr><tr><td>1</td><td>3.0</td><td>1.4</td></tr><tr><td>2</td><td>3.2</td><td>1.3</td></tr><tr><td>3</td><td>3.1</td><td>1.5</td></tr></table>		SepalWidthCm	PetalLengthCm	0	3.5	1.4	1	3.0	1.4	2	3.2	1.3	3	3.1	1.5			
	SepalWidthCm	PetalLengthCm																			
0	3.5	1.4																			
1	3.0	1.4																			
2	3.2	1.3																			
3	3.1	1.5																			
7	<code>dataset[dataset.columns[2:4]].iloc[5:10]</code>	in one Expression answer for the above two commands	<table><tr><th></th><th>SepalWidthCm</th><th>PetalLengthCm</th></tr><tr><td>5</td><td>3.9</td><td>1.7</td></tr><tr><td>6</td><td>3.4</td><td>1.4</td></tr><tr><td>7</td><td>3.4</td><td>1.5</td></tr><tr><td>8</td><td>2.9</td><td>1.4</td></tr><tr><td>9</td><td>3.1</td><td>1.5</td></tr></table>		SepalWidthCm	PetalLengthCm	5	3.9	1.7	6	3.4	1.4	7	3.4	1.5	8	2.9	1.4	9	3.1	1.5
	SepalWidthCm	PetalLengthCm																			
5	3.9	1.7																			
6	3.4	1.4																			
7	3.4	1.5																			
8	2.9	1.4																			
9	3.1	1.5																			

Checking of Missing Values in Dataset:

- `isnull()` is the function that is used to check missing values or null values in pandas python.
- `isna()` function is also used to get the count of missing values of column and row wise count of missing values
- The dataset considered for explanation is:

	Name	State	Gender	Score
0	George	Arizona	M	63.0
1	Andrea	Georgia	F	48.0
2	micheal	Newyork	M	56.0
3	maggie	Indiana	F	75.0
4	Ravi	Florida	M	NaN
5	Xien	California	M	77.0
6	Jalpa	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN

a. is there any missing values in dataframe as a whole Function: `DataFrame.isnull()`

Output:

	Name	State	Gender	Score
0	False	False	False	False
1	False	False	False	False
2	False	False	False	False
3	False	False	False	False
4	False	False	False	True
5	False	False	False	False
6	False	True	True	True
7	True	True	True	True

b. is there any missing values across each column Function: `DataFrame.isnull().any()`

Output:

```
Name      True
State     True
Gender    True
Score     True
dtype: bool
```

c. count of missing values across each column using isna() and isnull()

In order to get the count of missing values of the entire dataframe `isnull()` function is used. `sum()` which does the column wise sum first and doing another `sum()` will get the count of missing values of the entire dataframe.

Function: `dataframe.isnull().sum().sum()`

Output : 8

d. count row wise missing value using isnull()

Function: `dataframe.isnull().sum(axis = 1)`

Output:

```
0      0
1      0
2      0
3      0
4      1
5      0
6      3
7      4
dtype: int64
```

e. count Column wise missing value using isnull()

Method 1: Function: `dataframe.isnull().sum()`

Output:

```
Name      1
State     2
Gender    2
Score     3
dtype: int64
```

Method 2: Function: `dataframe.isna().sum()`

```
Name      1
State     2
Gender    2
Score     3
dtype: int64
```

f. count of missing values of a specific column.

Function: `dataframe.col_name.isnull().sum()` `df1.Gender.isnull().sum()`

Output: 2

g. groupby count of missing values of a column.

In order to get the count of missing values of the particular column by group in pandas we will be using `isnull()` and `sum()` function with `apply()` and `groupby()` which performs the group wise count of missing values as shown below.

Function:

```
df1.groupby(['Gender'])['Score'].apply(lambda x: x.isnull().sum())
```

Output:

```
Gender
F    0
M    1
Name: Score, dtype: int64
```

6. Panda functions for Data Formatting and Normalization

The Transforming data stage is about converting the data set into a format that can be analyzed or modelled effectively, and there are several techniques for this process.

a. Data Formatting: Ensuring all data formats are correct (e.g. object, text, floating number, integer, etc.) is another part of this initial 'cleaning' process. If you are working with dates in Pandas, they also need to be stored in the exact format to use special date-time functions. Functions used for data formatting

Sr. No	Data Frame Function	Description	Output
1.	df.dtypes	To check the data type	<pre>df.dtypes sepal length (cm) float64 sepal width (cm) float64 petal length (cm) float64 petal width (cm) float64 dtype: object</pre>
2.	df['petal length (cm)'] = df['petal length (cm)'].astype("int")	To change the data type (data type of 'petal length (cm)' changed to int)	<pre>df.dtypes sepal length (cm) float64 sepal width (cm) float64 petal length (cm) int64 petal width (cm) float64 dtype: object</pre>

b. Data normalization: Mapping all the nominal data values onto a uniform scale (e.g. from 0 to 1) is involved in data normalization. Making the ranges consistent across variables helps with statistical analysis and ensures better comparisons later on. It is also known as Min-Max scaling.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing

```
from sklearn import preprocessing
```

Step 2: Load the iris dataset in dataframe object df

Step 3: Print iris dataset.

```
df.head()
```

Step 5: Create a minimum and maximum processor object

```
min_max_scaler = preprocessing.MinMaxScaler()
```

Step 6: Separate the feature from the class label

```
x=df.iloc[:,4]
```

Step 6: Create an object to transform the data to fit minmax processor


```
x_scaled = min_max_scaler.fit_transform(x)
```

Step 7: Run the normalizer on the dataframe

```
df_normalized = pd.DataFrame(x_scaled)
```

Step 8: View the dataframe

```
df_normalized
```

Output: After Step 3

	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

Output after step 8:

	0	1	2	3
0	0.222222	0.625000	0.067797	0.041667
1	0.166667	0.416667	0.067797	0.041667
2	0.111111	0.500000	0.050847	0.041667
3	0.083333	0.458333	0.084746	0.041667
4	0.194444	0.666667	0.067797	0.041667

7. Panda Functions for handling categorical variables

- **Categorical variables** have values that describe a 'quality' or 'characteristic' of a data unit, like 'what type' or 'which category'.
- Categorical variables fall into **mutually exclusive (in one category or in another)** and **exhaustive (include all possible options)** categories. Therefore, categorical variables are qualitative variables and **tend to be represented by a non-numeric value**.
- Categorical features refer to **string type data** and can be easily understood by human beings. But in case of a machine, it cannot interpret the categorical data directly. Therefore, the categorical data must be **translated into numerical data that can be understood by machine**.

There are many ways to convert categorical data into numerical data. Here the three most used methods are discussed.

a. Label Encoding: Label Encoding refers to **converting the labels into a numeric form** so as to convert them into the machine-readable form. **It is an important preprocessing step for the structured dataset** in supervised learning. **Example :** Suppose we have a column Height in some dataset. After applying label encoding, the Height column is converted into:

Height	Height
Tall	0
Medium	1
Short	2

where 0 is the label for tall, 1 is the label for medium, and 2 is a label for short height. **Label Encoding on iris dataset:** For iris dataset the target column which is Species. It contains three species Iris-setosa, Iris-versicolor, Iris-virginica.

Sklearn Functions for Label Encoding:

- **preprocessing.LabelEncoder** : It Encode labels with value between 0 and n_classes-1.
- **fit_transform(y)**: **Parameters:** yarray-like of shape (n_samples,)

Target values. Returns: yarray-like of shape (n_samples,)

Encoded labels. This transformer should be used to encode target values, and not the input.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing
from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column.

df['Species'].unique()

output: array(['Iris-setosa', 'Iris-versicolor',
'Iris-virginica'], dtype=object)

Step 4: define label_encoder object knows how to understand word labels.

label_encoder = preprocessing.LabelEncoder()

Step 5: Encode labels in column 'species'.

df['Species']= label_encoder.fit_transform(df['Species'])

Step 6: Observe the unique values for the Species column.

df['Species'].unique()

Output: array([0, 1, 2], dtype=int64)

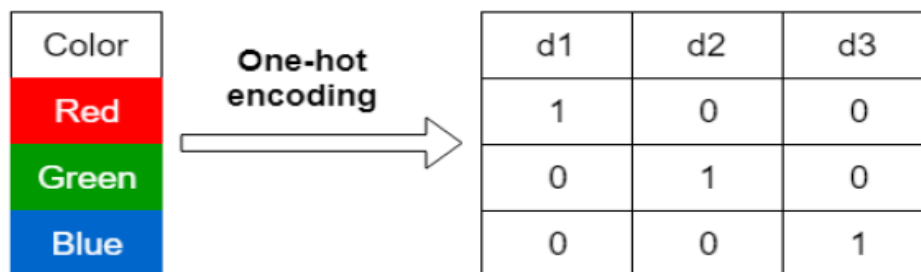
- Use LabelEncoder when there are only two possible values of a categorical feature. For example, features having value such as yes or no. Or, maybe, gender features when there are only two possible values including male or female.

Limitation: Label encoding converts the data in machine-readable form, but it assigns a **unique number (starting from 0) to each class of data**. This may lead to the generation of **priority issues in the data sets**. A label with a high value may be considered to have high priority than a label having a lower value.

b. One-Hot Encoding:

In one-hot encoding, we create a new set of dummy (binary) variables that is equal to the number of categories (k) in the variable. For example, let's say we have a categorical variable Color with three categories called "Red",

“Green” and “Blue”, we need to use three dummy variables to encode this variable using one-hot encoding. A dummy (binary) variable just takes the value 0 or 1 to indicate the exclusion or inclusion of a category.



In one-hot encoding,

“Red” color is encoded as [1 0 0] vector of size 3.

“Green” color is encoded as [0 1 0] vector of size 3.

“Blue” color is encoded as [0 0 1] vector of size 3.

One-hot encoding on iris dataset: For iris dataset the target column which is Species. It contains three species Iris-setosa, Iris-versicolor, Iris-virginica.

Sklearn Functions for One-hot Encoding:

- **sklearn.preprocessing.OneHotEncoder():** Encode categorical integer features using a one-hot aka one-of-K scheme

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing
from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column.
df['Species'].unique()

output: array(['Iris-setosa', 'Iris-versicolor',
'Iris-virginica'], dtype=object)

Step 4: Apply label_encoder object for label encoding the Observe the unique values for the Species column.

df['Species'].unique()

Output: array([0, 1, 2], dtype=int64)

Step 5: Remove the target variable from dataset

features_df=df.drop(columns=['Species'])

Step 6: Apply one_hot encoder for Species column.

enc = preprocessing.OneHotEncoder()

enc_df=pd.DataFrame(enc.fit_transform(df[['Species']])).toarray()

Step 7: Join the encoded values with Features variable

df_encode = features_df.join(enc_df)

Step 8: Observe the merge dataframe

df_encode

Step 9: Rename the newly encoded columns.

df_encode.rename(columns = {0:'Iris-Setosa',
1:'Iris-Versicolor',2:'Iris-virginica'}, inplace = True)

Step 10: Observe the merge dataframe

df_encode

Output after Step 8:

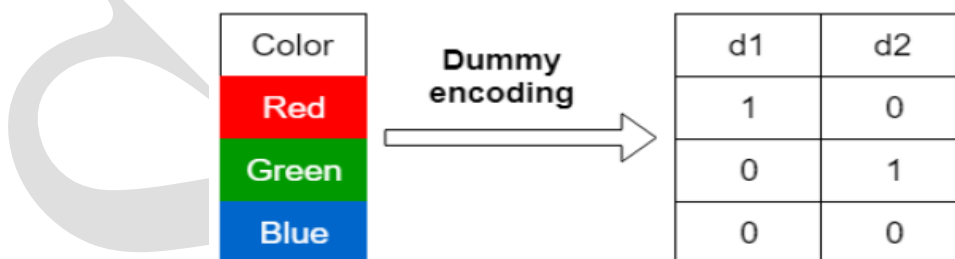
	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	0	1	2
0	5.1	3.5	1.4	0.2	1.0	0.0	0.0
1	4.9	3.0	1.4	0.2	1.0	0.0	0.0
2	4.7	3.2	1.3	0.2	1.0	0.0	0.0
3	4.6	3.1	1.5	0.2	1.0	0.0	0.0
4	5.0	3.6	1.4	0.2	1.0	0.0	0.0

Output after Step 10:

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Iris-Setosa	Iris-Versicolor	Iris-virginica
0	5.1	3.5	1.4	0.2	1.0	0.0	0.0
1	4.9	3.0	1.4	0.2	1.0	0.0	0.0
2	4.7	3.2	1.3	0.2	1.0	0.0	0.0
3	4.6	3.1	1.5	0.2	1.0	0.0	0.0
4	5.0	3.6	1.4	0.2	1.0	0.0	0.0

c. Dummy Variable Encoding:

Dummy encoding also uses dummy (binary) variables. Instead of creating a number of dummy variables that is equal to the number of categories (k) in the variable, dummy encoding uses k-1 dummy variables. To encode the same Color variable with three categories using the dummy encoding, we need to use only two dummy variables.



In dummy encoding,

“Red” color is encoded as [1 0] vector of size 2.

“Green” color is encoded as [0 1] vector of size 2.

“Blue” color is encoded as [0 0] vector of size 2.

Dummy encoding removes a duplicate category present in the one-hot encoding.

Pandas Functions for One-hot Encoding with dummy variables:

- **pandas.get_dummies(data, prefix=None, prefix_sep='_', dummy_na=False, columns=None, sparse=False, drop_first=False, dtype=None):** Convert categorical variable into dummy/indicator variables.

- **Parameters:**

data: array-like, Series, or DataFrame

Data of which to get dummy indicators.

prefixstr: list of str, or dict of str, default None

String to append DataFrame column names.

prefix_sep: str, default '_'

If appending prefix, separator/delimiter to use. Or pass a list or dictionary as with prefix.

dummy_na: bool, default False

Add a column to indicate NaNs, if False NaNs are ignored.

columns: list-like, default None

Column names in the DataFrame to be encoded. If columns is None then all the columns with object or category dtype will be converted.

sparse: bool, default False

Whether the dummy-encoded columns should be backed by a SparseArray (True) or a regular NumPy array (False).

drop_first: bool, default False

Whether to get k-1 dummies out of k categorical levels by removing the first level.

dtype: dtype, default np.uint8

Data type for new columns. Only a single dtype is allowed.

- **Return :** DataFrame with Dummy-coded data.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing
from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column.

df['Species'].unique()

output: array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)

Step 4: Apply label_encoder object for label encoding the Observe the unique values for the Species column. df['Species'].unique()

Output: array([0, 1, 2], dtype=int64)

Step 6: Apply one_hot encoder with dummy variables for Species column.

one_hot_df = pd.get_dummies(df, prefix="Species",
columns=['Species'], drop_first=True)

Step 7: Observe the merge dataframe

one_hot_df

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species_1	Species_2
0	5.1	3.5	1.4	0.2	0	0
1	4.9	3.0	1.4	0.2	0	0
2	4.7	3.2	1.3	0.2	0	0
3	4.6	3.1	1.5	0.2	0	0
4	5.0	3.6	1.4	0.2	0	0

Conclusion: In this way we have explored the functions of the python library for Data Preprocessing, Data Wrangling Techniques and How to Handle missing values on Iris Dataset.

Questions:

1. Explain Data Frame with Suitable example.
2. What is the limitation of the label encoding method?
3. What is the need of data normalization?
4. What are the different Techniques for Handling the Missing Data?

SCOPE

Assignment No.	2 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 2

Title: Data Wrangling-II

Create an “Academic performance” dataset of students and perform the following operations using Python.

1. Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them.
2. Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them.
3. Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable, to convert a non-linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution. Reason and document your approach properly.

Prerequisites:

1. Basic of Python Programming
2. Concept of Data Preprocessing, Data Formatting, Data Normalization and Data Cleaning.

Objectives: Students should be able to perform the data wrangling operation using Python on any open source dataset

Theory:

1. Creation of Dataset using Microsoft Excel.

The dataset is created in “CSV” format.

- The name of dataset is **StudentsPerformance**
- **The features of the dataset are:** Math_Score, Reading_Score, Writing_Score, Placement_Score, Club_Join_Date .
- **Number of Instances:** 30
- **The response variable is:** Placement_Offer_Count .
- **Range of Values:**

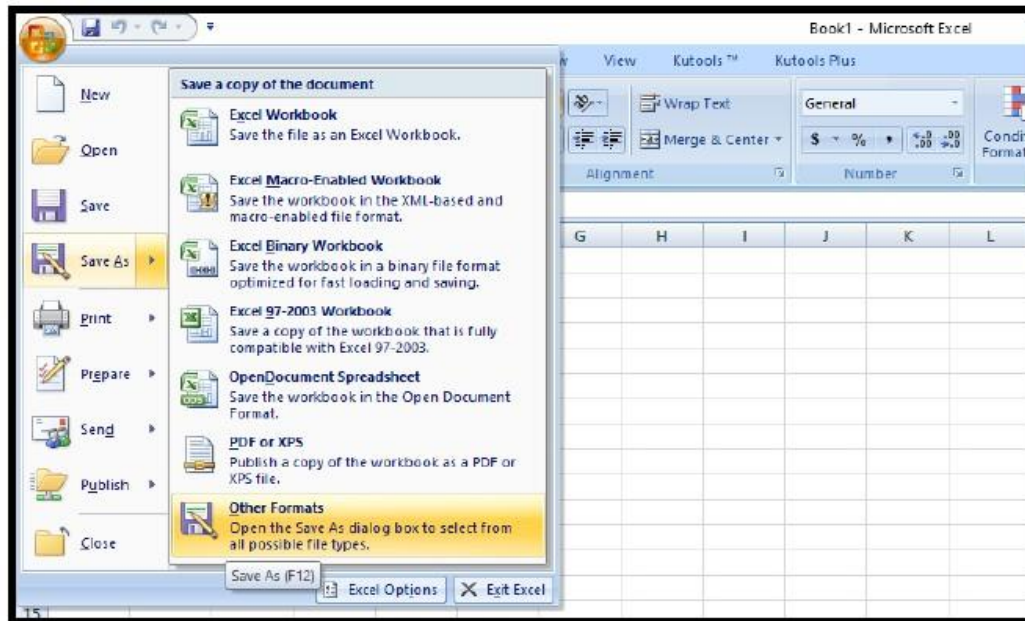
Math_Score [60-80], Reading_Score[75-,95], ,Writing_Score [60,80], Placement_Score[75-100], Club_Join_Date [2018-2021].

- **The response variable is** the number of placement offers facilitated to particular students, which is largely depend on Placement_Score To fill the values in the dataset the **RANDBETWEEN** is used. Returns a random integer number between the numbers you specify.

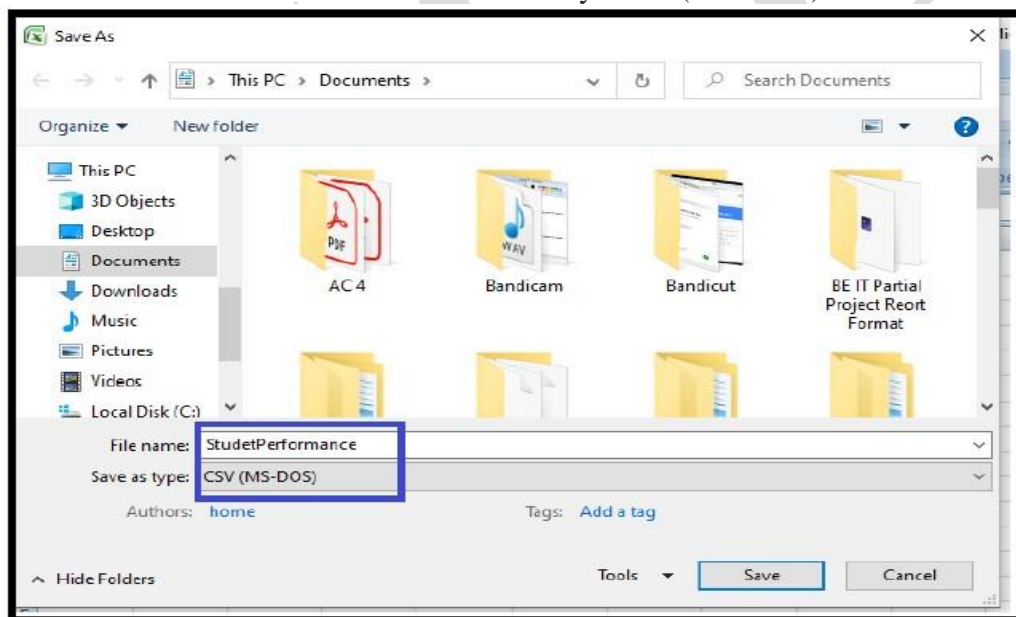
Syntax : RANDBETWEEN(bottom, top) **Bottom** The smallest integer and **Top** The largest integer RANDBETWEEN will return. For better understanding and visualization, 20% impurities are added into each variable to the dataset.

The step to create the dataset are as follows:

Step 1: Open Microsoft Excel and click on Save As. Select Other Formats



Step 2: Enter the name of the dataset and Save the dataset as type CSV(MS-DOS).



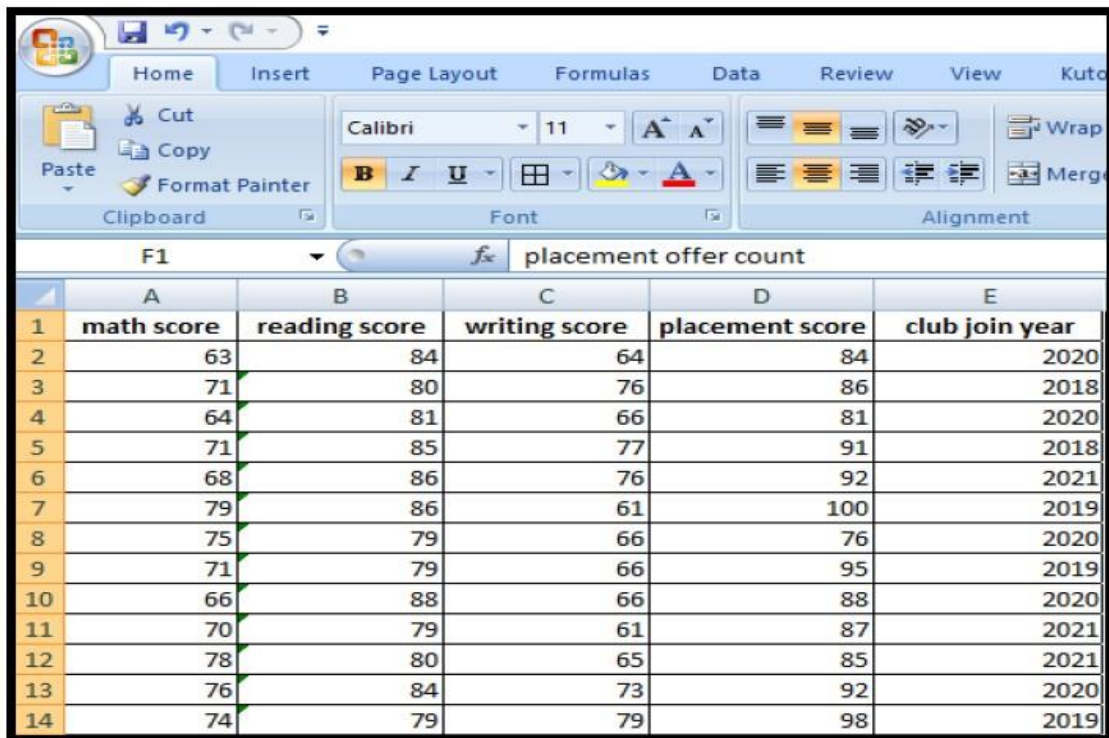
Step 3: Enter the name of features as column header.

	A	B	C	D	E	F
1	math score	reading score	writing score	placement score	club join year	placement offer count
2						
3						
4						
5						
6						
7						

Step 3: Fill the data by using **RANDBETWEEN** function. For every feature, fill the data by considering above specified range. one example is given.

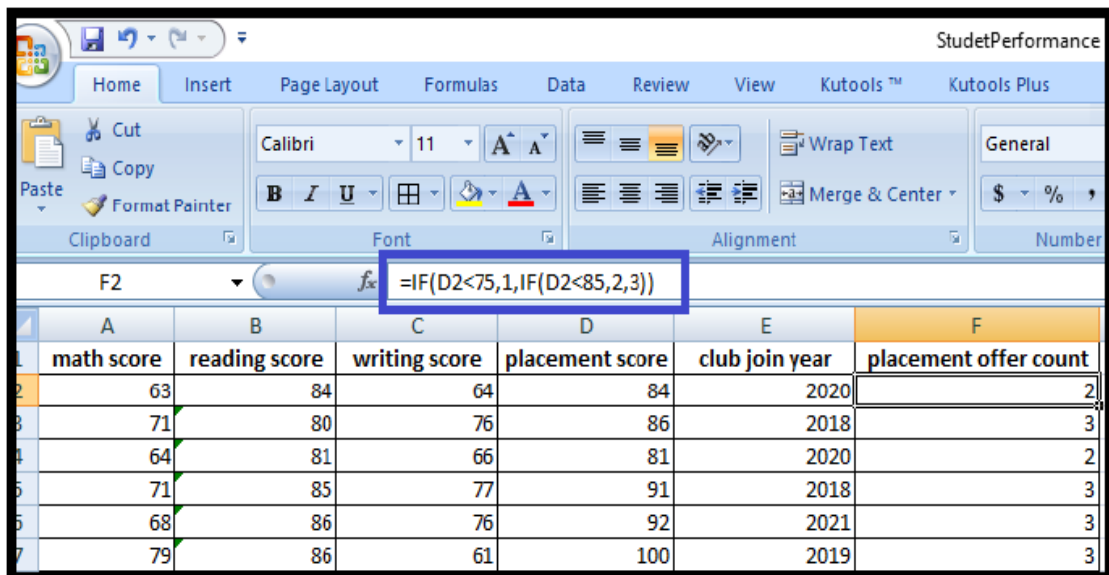
	A	B	C	D	E	F
1	math score	reading score	writing score	placement score	club join year	placement offer count
2	VEEN(60,80)					
3						

Scroll down the cursor for 30 rows to create 30 instances. Repeat this for the features, Reading_Score, Writing_Score, Placement_Score, Club_Join_Date.



	A	B	C	D	E
	math score	reading score	writing score	placement score	club join year
1					
2	63	84	64	84	2020
3	71	80	76	86	2018
4	64	81	66	81	2020
5	71	85	77	91	2018
6	68	86	76	92	2021
7	79	86	61	100	2019
8	75	79	66	76	2020
9	71	79	66	95	2019
10	66	88	66	88	2020
11	70	79	61	87	2021
12	78	80	65	85	2021
13	76	84	73	92	2020
14	74	79	79	98	2019

The placement count largely depends on the placement score. It is considered that if placement score <75, 1 offer is facilitated; for placement score >75, 2 offer is facilitated and for else (>85) 3 offer is facilitated. Nested If formula is used for ease of data filling.



	A	B	C	D	E	F
	math score	reading score	writing score	placement score	club join year	placement offer count
1						
2	63	84	64	84	2020	2
3	71	80	76	86	2018	3
4	64	81	66	81	2020	2
5	71	85	77	91	2018	3
6	68	86	76	92	2021	3
7	79	86	61	100	2019	3

Step 4: In 20% data, fill the impurities. The range of math score is [60,80], updating a few instances values below 60 or above 80. Repeat this for Writing_Score [60,80], Placement_Score[75-100], Club_Join_Date [2018-2021].

	A	B	C	D	E
1	math score	reading score	writing score	placement score	club join year
2	68	94	64	90	2018
3	72	85	70	86	2018
4	94	90	64	91	2020

Step 5: To violate the rule of response variable, update few value. If placement score is greater than 85, facilitated only 1 offer.

	A	B	C	D	E	F
1	math score	reading score	writing score	placement score	club join year	placement offer count
2	70	91	64	87	2019	3
3	77	75	67	81	2020	2
4	94	84	73	99	2019	3
5	78	84	77	96	2020	1

The dataset is created with the given description

2. Identification and Handling of Null Values

Missing Data can occur when no information is provided for one or more items or for a whole unit. Missing Data is a very big problem in real-life scenarios. Missing Data can also refer to as NA(Not Available) values in pandas. In DataFrame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed. For Example, Suppose different users being surveyed may choose not to share their income, some users may choose not to share the address in this way many datasets went missing.

In Pandas missing data is represented by two value:

1. **None:** None is a Python singleton object that is often used for missing data in Python code.
2. **NaN :** NaN (an acronym for Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation. Pandas treat None and NaN as essentially interchangeable for indicating missing or null values. To facilitate this convention, there are several useful functions for detecting, removing, and replacing null values in Pandas DataFrame :

- isnull()
- notnull()
- dropna()
- fillna()
- replace()

1. Checking for missing values using isnull() and notnull()

• Checking for missing values using isnull()

In order to check null values in Pandas DataFrame, isnull() function is used. This function return dataframe of Boolean values which are True for NaN values.

• Checking for missing values using notnull()

In order to check null values in Pandas Dataframe, notnull() function is used. This function return dataframe of Boolean values which are False for NaN values.

2. Filling missing values using dropna(), fillna(), replace()

In order to fill null values in a datasets, fillna(), replace() functions are used.

These functions replace NaN values with some value of their own. All these functions help in filling null values in datasets of a DataFrame.

- For replacing null values with NaN `missing_values = ["Na", "na"]`

Filling a null values using `replace()` method

Following line will replace Nan value in dataframe with value -99 `ndf.replace(to_replace = np.nan, value = -99)`

Deleting null values using `dropna()` method

In order to drop null values from a dataframe, `dropna()` function is used. This function drops Rows/Columns of datasets with Null values in different ways.

1. Dropping rows with at least 1 null value
2. Dropping rows if all values in that row are missing
3. Dropping columns with at least 1 null value.
4. Dropping Rows with at least 1 null value in CSV file

3. Identification and Handling of Outliers

3.1 Identification of Outliers

One of the most important steps as part of data pre processing is detecting and treating the outliers as they can negatively affect the statistical analysis and the training process of a machine learning algorithm resulting in lower accuracy.

○ 1. What are Outliers?

We all have heard of the idiom 'odd one out' which means something unusual in comparison to the others in a group. Similarly, an Outlier is an observation in a given dataset that lies far from the rest of the observations. That means an outlier is vastly larger or smaller than the remaining values in the set.

○ 2. Why do they occur?

An outlier may occur due to the variability in the data, or due to experimental error/human error. They may indicate an experimental error or heavy skewness in the data (heavy-tailed distribution).

○ 3. What do they affect?

In statistics, we have three measures of central tendency namely Mean, Median, and Mode. They help us describe the data. Mean is the accurate measure to describe the data when we do not have any outliers present. Median is used if there is an outlier in the dataset. Mode is used if there is an outlier AND about ½ or more of the data is the same. 'Mean' is the only measure of central tendency that is affected by the outliers which in turn impacts Standard deviation.

■ Example:

Consider a small dataset, `sample= [15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]`. By looking at it, one can quickly say '101' is an outlier that is much larger than the other values.

with outlier	without outlier
Mean: 20.08	Mean: 12.72
Median: 14.0	Median: 13.0
Mode: 15	Mode: 15
Variance: 614.74	Variance: 21.28
Std dev: 24.79	Std dev: 4.61

From the above calculations, we can clearly say the Mean is more affected than the Median.

4. Detecting Outliers

If our dataset is small, we can detect the outlier by just looking at the dataset. But what if we have a huge dataset, how do we identify the outliers then? We need to use visualization and mathematical techniques. Below are some of the techniques of detecting outliers

- Boxplots
- Scatterplots
- Z-score
- Inter Quantile Range(IQR)

4.1 Detecting outliers using Boxplot:

It captures the summary of the data effectively and efficiently with only a simple box and whiskers. Boxplot summarizes sample data using 25th, 50th, and 75th percentiles. One can just get in sights (quartiles, median, and outliers) into the dataset by just looking at its boxplot.

4.2 Detecting outliers using Scatterplot:

It is used when you have paired numerical data, or when your dependent variable has multiple values for each reading independent variable, or when trying to determine the relationship between the two variables. In the process of utilizing the scatter plot, one can also use it for outlier detection. To plot the scatter plot one requires two variables that are somehow related to each other. So here Placement score and Placement count features are used.

4.3 Detecting outliers using Z-Score:

Z-Score is also called a standard score. This value/score helps to understand how far is the data point from the mean. And after setting up a threshold value one can utilize z score values of data points to define the outliers.

$Zscore = (data_point - mean) / std. deviation.$

4.4 Detecting outliers using Inter Quantile Range(IQR):

IQR (Inter Quartile Range) Inter Quartile Range approach to finding the outliers is the most commonly used and most trusted approach used in the research field.

$IQR = Quartile3 - Quartile1$

To define the outlier base value is defined above and below datasets normal range namely Upper and Lower bounds, define the upper and the lower

bound ($1.5 * IQR$ value is considered) :

upper = $Q3 + 1.5 * IQR$

lower = $Q1 - 1.5 * IQR$

In the above formula as according to statistics, the 0.5 scale-up of IQR

($new_IQR = IQR + 0.5 * IQR$) is taken.

3.2 Handling of Outliers:

For removing the outlier, one must follow the same process of removing an entry from the dataset using its exact position in the dataset because in all the above methods of detecting the outliers end result is the list of all those data items that satisfy the outlier definition according to the method used. Below are some of the methods of treating the outliers

- Trimming/removing the outlier
- Quantile based flooring and capping
- Mean/Median imputation

- **Trimming/removing the outlier:**

In this technique, we remove the outliers from the dataset.

- **Quantile based flooring and capping:**

In this technique, the outlier is capped at a certain value above the 90th percentile value or floored at a factor below the 10th percentile value

- **Mean/Median imputation:**

As the mean value is highly influenced by the outliers, it is advised to replace the outliers with the median value.

4. Data Transformation for the purpose of :

Data transformation is the process of converting raw data into a format or structure that would be more suitable for model building and also data discovery in general. The process of data transformation can also be referred to as extract/transform/load (ETL). The extraction phase involves identifying and pulling data from the various source systems that create data and then moving the data to a single repository. Next, the raw data is cleansed, if needed. It's then transformed into a target format that can be fed into operational systems or into a data warehouse, a data lake or another repository for use in business intelligence and analytics applications. The transformation The data are transformed in ways that are ideal for mining the data. The data transformation involves steps that are.

- **Smoothing:** It is a process that is used to remove noise from the dataset using some algorithms. It allows for highlighting important features present in the dataset. It helps in predicting the patterns

- **Aggregation:** Data collection or aggregation is the method of storing and presenting data in a summary format. The data may be obtained from multiple data sources to integrate these data sources into a data analysis description. This is a crucial step since the accuracy of data analysis insights is highly dependent on the quantity and quality of the data used.

- **Generalization:** It converts low-level data attributes to high-level data attributes using concept hierarchy. For Example Age initially in Numerical form (22, 25) is converted into categorical value (young, old).

- **Normalization:** Data normalization involves converting all data variables into a given range. Some of the techniques that are used for accomplishing normalization are:

- **Min-max normalization:** This transforms the original data linearly.

- **Z-score normalization:** In z-score normalization (or zero-mean normalization) the values of an attribute (A), are normalized based on the mean of A and its standard deviation.

- **Normalization by decimal scaling:** It normalizes the values of an attribute by changing the position of their decimal points

Attribute or feature construction.

- **New attributes constructed from the given ones:** Where new attributes are created & applied to assist the mining process from the given set of attributes. This simplifies the original data & makes the mining more efficient. In this assignment, The purpose of this transformation should be one of the following reasons:

- a. To change the scale for better understanding (Attribute or feature construction)**

Here the Club_Join_Date is transferred to Duration.

Algorithm:

Step 1 : Import pandas and numpy libraries

```
import pandas as pd
```

```
import numpy as np
```

Step 2: Load the dataset in dataframe object df

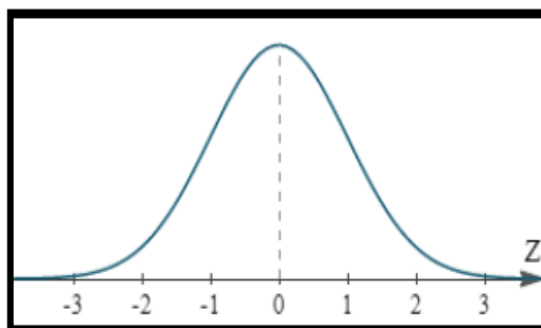
```
df=pd.read_csv("/content/demo.csv")
```

Step 3: Display the data frame

**b. To decrease the skewness and convert distribution into normal distribution
(Normalization by decimal scaling)**

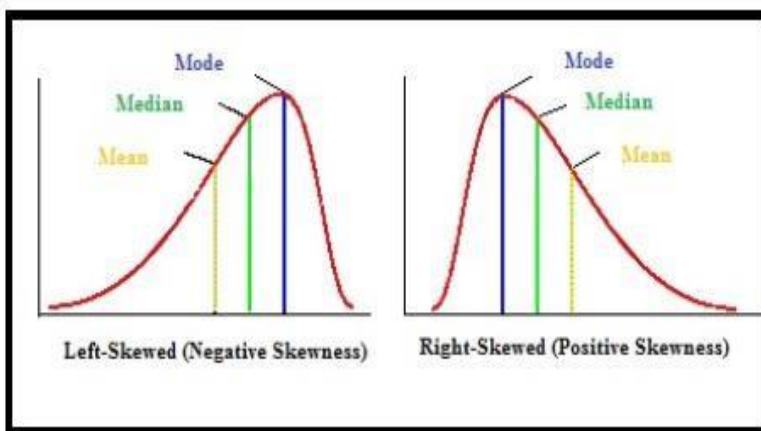
Data Skewness: It is asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right. Skewness can be quantified to define the extent to which a distribution differs from a normal distribution.

Normal Distribution: In a normal distribution, the graph appears as a classical, symmetrical “bell-shaped curve.” The mean, or average, and the mode, or maximum point on the curve, are equal.

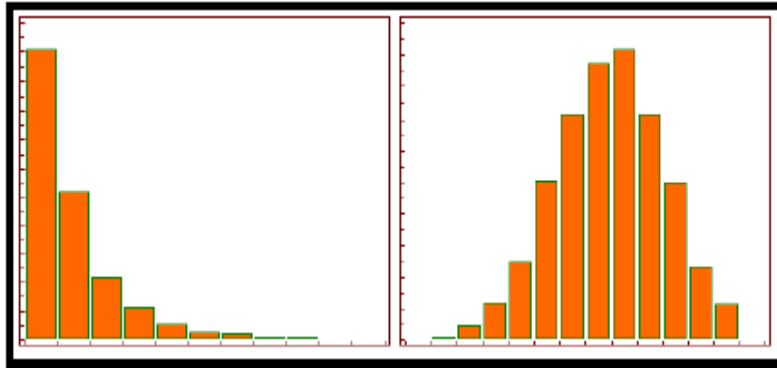


A positively skewed distribution means that the extreme data results are larger. This skews the data in that it brings the mean (average) up. The mean will be larger than the median in a Positively skewed distribution.

A negatively skewed distribution means the opposite: that the extreme data results are smaller. This means that the mean is brought down, and the median is larger than the mean in a negatively skewed distribution.



Reducing skewness A data transformation may be used to reduce skewness. A distribution that is symmetric or nearly so is often easier to handle and interpret than a skewed distribution. The logarithm, x to log base 10 of x , or x to log base e of x ($\ln x$), or x to log base 2 of x , is a strong transformation with a major effect on distribution shape. It is commonly used for reducing right skewness and is often appropriate for measured variables. It can not be applied to zero or negative values.



Conclusion: In this way we have explored the functions of the python library for Data Identifying and handling the outliers. Data Transformations Techniques are explored with the purpose of creating the new variable and reducing the skewness from datasets.

Questions:

1. Explain the methods to detect the outlier.
2. Explain data transformation methods
3. Write the algorithm to display the statistics of Null values present in the dataset.
4. Write an algorithm to replace the outlier value with the mean of the variable.

Assignment No.	3 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 3

Title: Descriptive Statistics - Measures of Central Tendency and variability

Perform the following operations on any open source dataset (e.g., data.csv)

1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.
2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris- versicolor' of iris.csv dataset.

Prerequisites: 1. Basic of Python Programming

2. Concept of statistics such as mean, median, minimum, maximum, standard deviation etc.

Objectives: Students should be able to perform the Statistical operations using Python on any open source dataset.

Theory:**What is Statistics?**

Statistics is the science of collecting data and analyzing them to infer proportions (sample) that are representative of the population. In other words, statistics is interpreting data in order to make predictions for the population.

Branches of Statistics:

There are two branches of Statistics.

DESCRIPTIVE STATISTICS : Descriptive Statistics is a statistics or a measure that describes the data.

INFERENTIAL STATISTICS : Using a random sample of data taken from a population to describe and make inferences about the population is called Inferential Statistics.

Descriptive Statistics

Descriptive Statistics is summarizing the data at hand through certain numbers like mean, median etc. so as to make the understanding of the data easier. It does not involve any generalization or inference beyond what is available. This means that the descriptive statistics are just the representation of the data (sample) available and not based on any theory of probability.

Commonly Used Measures

1. Measures of Central Tendency
2. Measures of Dispersion (or Variability)

• Measures of Central Tendency

A Measure of Central Tendency is a one number summary of the data that typically describes the Centre of the data. This one number summary is of three types.

a. **Mean :** Mean is defined as the ratio of the sum of all the observations in the data to the total number of observations. This is also known as Average. Thus mean is a number around which the entire data set is spread.

Consider the following data points. 17, 16, 21, 18, 15, 17, 21, 19, 11, 23

$$\text{Mean} = \frac{17 + 16 + 21 + 18 + 15 + 17 + 21 + 19 + 11 + 23}{10} = \frac{178}{10} = 17.8$$

b. **Median** : Median is the point which divides the entire data into two equal halves. One-half of the data is less than the median, and the other half is greater than the same. Median is calculated by first arranging the data in either ascending or descending order.

- If the number of observations is odd, the median is given by the middle observation in the sorted form.
- If the number of observations are even, median is given by the mean of the two middle observations in the sorted form. An important point to note is that the order of the data (ascending or descending) does not affect the median. To calculate Median, let's arrange the data in ascending order. 11, 15, 16, 17, 17, 18, 19, 21, 21, 23 Since the number of observations is even (10), median is given by the average of the two middle observations (5th and 6th here).

$$\text{Median} = \frac{5^{\text{th}} \text{ Obs} + 6^{\text{th}} \text{ Obs}}{2} = \frac{17 + 18}{2} = 17.5$$

c. **Mode** : Mode is the number which has the maximum frequency in the entire data set, or in other words, mode is the number that appears the maximum number of times. A data can have one or more than one mode.

- If there is only one number that appears the maximum number of times, the data has one mode, and is called Uni-modal.
- If there are two numbers that appear the maximum number of times, the data has two modes, and is called Bi-modal.
- If there are more than two numbers that appear the maximum number of times, the data has more than two modes, and is called Multi-modal. Consider the following data points. 17, 16, 21, 18, 15, 17, 21, 19, 11, 23

Mode is given by the number that occurs the maximum number of times. Here, 17 and 21 both occur twice. Hence, this is a Bimodal data and the modes are 17 and 21.

● Measures of Dispersion (or Variability)

Measures of Dispersion describes the spread of the data around the central value (or the Measures of Central Tendency)

1. **Absolute Deviation from Mean** — The Absolute Deviation from Mean, also called Mean Absolute Deviation (MAD), describes the variation in the data set, in the sense that it tells the average absolute distance of each data point in the set. It is calculated as

$$\text{Mean Absolute Deviation} = \frac{1}{N} \sum_{i=1}^N |X_i - \bar{X}|$$

2. **Variance** — Variance measures how far are data points spread out from the mean. A high variance indicates that data points are spread widely and a small variance indicates that the data points are closer to the mean of the data set. It is calculated as

$$\text{Variance} = \frac{1}{N} \sum_{i=1}^N (X_i - \bar{X})^2$$

3. **Standard Deviation** — The square root of Variance is called the Standard Deviation. It is calculated as

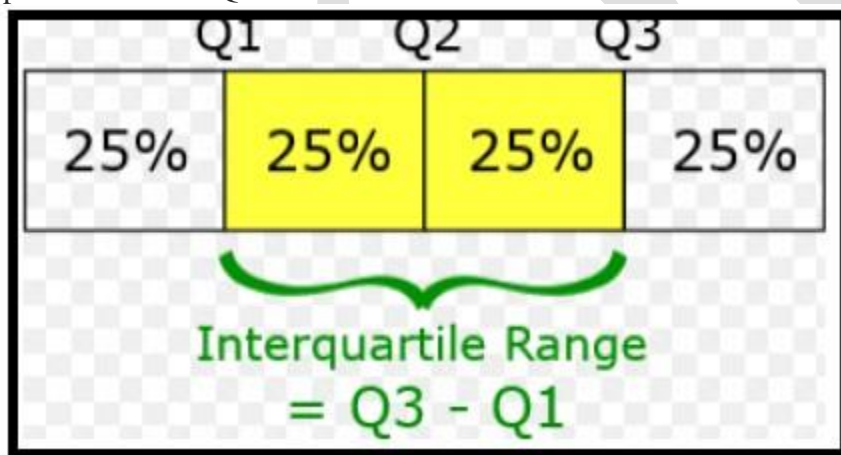
$$\text{Std Deviation} = \sqrt{\text{Variance}} = \sqrt{\frac{1}{N} \sum_{i=1}^N (X_i - \bar{X})^2}$$

4. **Range** — Range is the difference between the Maximum value and the Minimum value in the data set. It is given as

$$\text{Range} = \text{Maximum} - \text{Minimum}$$

5. **Quartiles** — Quartiles are the points in the data set that divides the data set into four equal parts. Q1, Q2 and Q3 are the first, second and third quartile of the data set.

- 25% of the data points lie below Q1 and 75% lie above it.
- 50% of the data points lie below Q2 and 50% lie above it. Q2 is nothing but Median.
- 75% of the data points lie below Q3 and 25% lie above it.

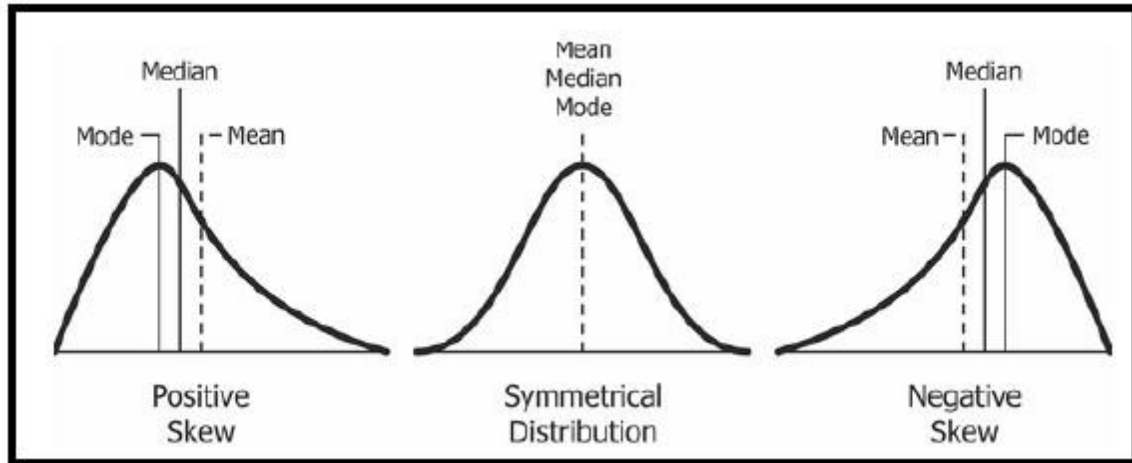


6. **Skewness** — The measure of asymmetry in a probability distribution is defined by Skewness. It can either be positive, negative or undefined.

$$\text{Skewness} = \frac{3 (\text{Mean} - \text{Median})}{\text{Std Deviation}}$$

Positive Skew — This is the case when the tail on the right side of the curve is bigger than that on the left side. For these distributions, mean is greater than the mode.

Negative Skew — This is the case when the tail on the left side of the curve is bigger than that on the right side. For these distributions, mean is smaller than the mode. The most commonly used method of calculating Skewness is If the skewness is zero, the distribution is symmetrical. If it is negative, the distribution is Negatively Skewed and if it is positive, it is Positively Skewed.

**Conclusion:**

Descriptive statistics describes the characteristics of a data set. Descriptive statistics consists of two basic categories of measures: • measures of central tendency and • measures of variability (or spread).

Measures of central tendency describe the Centre of a data set. It includes the mean, median, and mode.

Measures of variability or spread describe the dispersion of data within the set and it includes standard deviation, variance, minimum and maximum variables.

Questions:

1. Explain Measures of Central Tendency with examples.
2. What are the different types of variables? Explain with examples.
3. Which method is used to display statistics of the data frame? write the code.

Assignment No.	4 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 4

Title: Data Analytics I

Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (<https://www.kaggle.com/c/boston-housing>). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset.

Prerequisites: 1. Basic of Python Programming
2. Concept of Regression.

Objectives: Students should be able to data analysis using liner regression using Python for any open source dataset.

Theory:

1. **Linear Regression:** It is a machine learning algorithm based on supervised learning. It targets prediction values on the basis of independent variables.

- It is preferred to find out the relationship between forecasting and variables.
- A linear relationship between a dependent variable (Y) is continuous; while independent variable (X) relationship may be continuous or discrete. A linear relationship should be available in between predictor and target variable so known as Linear Regression.
- Linear regression is popular because the cost function is Mean Squared Error (MSE) which is equal to the average squared difference between an observation's actual and predicted values.
- It is shown as an equation of line like : $Y = m \cdot X + b + e$

Where : b is intercepted, m is slope of the line and e is error term. This equation can be used to predict the value of target variable Y based on given predictor variable(s) X, as shown in Fig. 1.

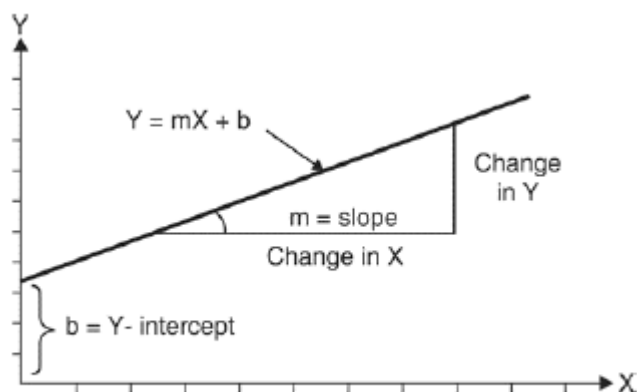
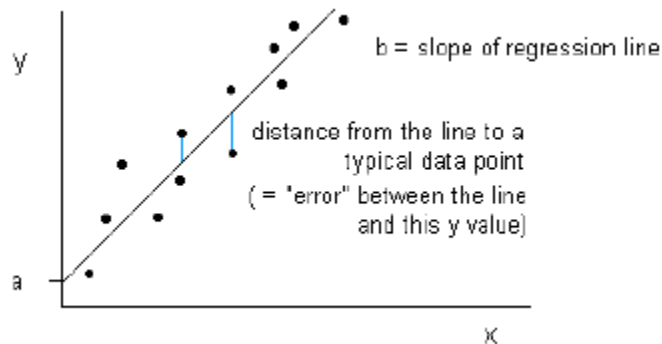


Fig. 1: geometry of linear regression

- Fig. 2 shown below is about the relation between weight (in Kg) and height (in cm), a linear relation. It is an approach of studying in a statistical manner to summarize and learn the relationships among continuous (quantitative) variables.
- Here a variable, denoted by 'x' is considered as the predictor, explanatory, or independent variable.

- Another variable, denoted 'y', is considered as the response, outcome, or dependent variable. While "predictor" and "response" used to refer to these variables.
- Simple linear regression technique concerned with the study of only one predictor variable.

Fig.2 : Relation between weight (in Kg) and height (in cm)



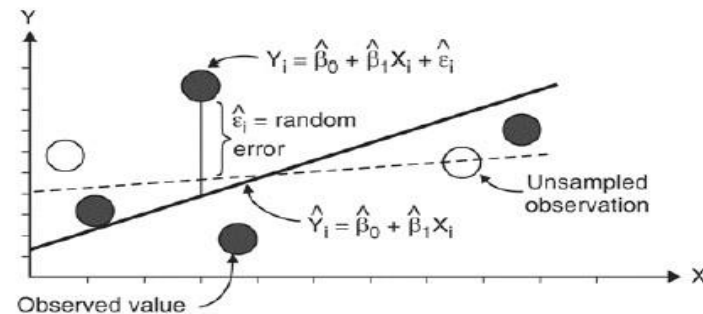
Multi Variate Regression :It concerns the study of two or more predictor variables. Usually a transformation of the original features into polynomial features from a given degree is preferred and further Linear Regression is applied on it.

- A simple linear model $Y = a + bX$ in original feature will be transformed into polynomial feature is transformed and further a linear regression applied to it and it will be something like $Y = a + bX + cX^2$
- If a high degree value is used in transformation the curve becomes over-fitted as it captures the noise from data as well.

2. Least Square Method for Linear Regression

- Linear Regression involves establishing linear relationships between dependent and independent variables. Such a relationship is portrayed in the form of an equation also known as the linear model.
 - A simple linear model is the one which involves only one dependent and one independent variable. Regression Models are usually denoted in Matrix Notations.
 - However, for a simple univariate linear model, it can be denoted by the regression equation $y^{\wedge} = \beta_0 + \beta_1 x$ ----- (1)
- where y^{\wedge} is the dependent or the response variable x is the independent or the input variable β_0 is the value of y when $x=0$ or the y intercept β_1 is the value of slope of the line ϵ is the error or the noise
- This linear equation represents a line also known as the 'regression line'. The least square estimation technique is one of the basic techniques used to guess the values of the parameters and based on a sample set.
 - This technique estimates parameters β_0 and β_1 and by trying to minimize the square of errors at all the points in the sample set. The error is the deviation of the actual sample
 - data point from the regression line. The technique can be represented by the equation.

$$\min \sum_{i=0}^n (\hat{y} - y)^2 \quad (2)$$



Where,

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

Population Y-intercept Population slope Random error

Dependent (Response) variable Independent (Explanatory) variable

Using differential calculus on equation 1 we can find the values of β_0 and such β_1 that the sum of squares (that is equation 2) is minimum.

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \quad (3)$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x} \quad (4)$$

Once the Linear Model is estimated using equations (3) and (4), we can estimate the value of the dependent variable in the given range only. Going outside the range is called extrapolation which is inaccurate if simple regression techniques are used.

3. Measuring Performance of Linear Regression

Mean Square Error:

The Mean squared error (MSE) represents the error of the estimator or predictive model created based on the given set of observations in the sample. Two or more regression models created using a given sample data can be compared based on their MSE. The lesser the MSE, the better the regression model is. When the linear regression model is trained using a given set of observations, the model with the least mean sum of squares error (MSE) is selected as the best model. The Python or R packages select the best-fit model as the model with the lowest MSE or lowest RMSE when training the linear regression models. Mathematically, the MSE can be calculated as the average sum of the squared difference between the actual value and the predicted or estimated value represented by the regression model (line or plane).

$$MSE = \frac{1}{n} \sum \left(y - \hat{y} \right)^2$$

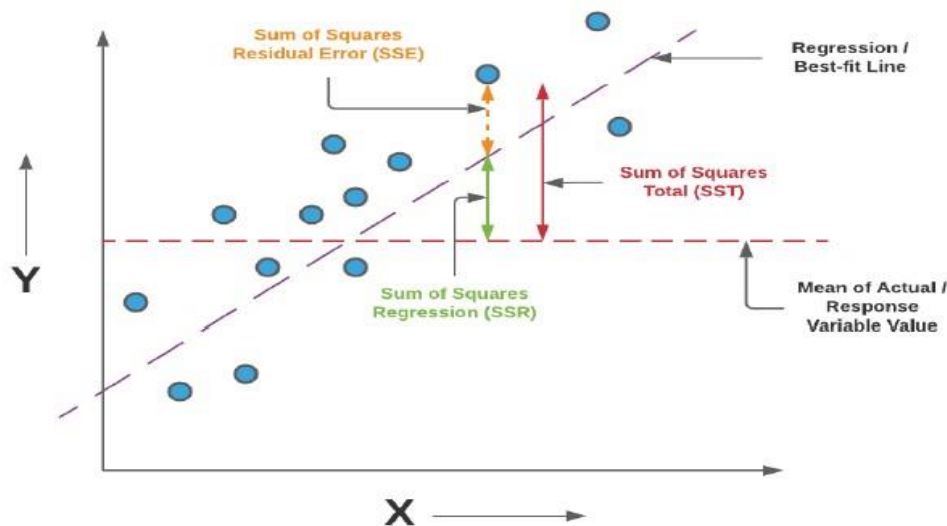
The square of the difference
between actual and
predicted

An MSE of zero (0) represents the fact that the predictor is a perfect predictor. RMSE:

Root Mean Squared Error method that basically calculates the least-squares error and takes a root of the summed values. Mathematically speaking, Root Mean Squared Error is the square root of the sum of all errors divided by the total number of values. This is the formula to calculate RMSE

$$\text{RMSE} = \sqrt{\sum_{i=1}^n \frac{1}{n} (\hat{y}_i - y_i)^2}$$

RMSE - Least Squares Regression Method – Edureka R-Squared :



R-Squared is the ratio of the sum of squares regression (SSR) and the sum of squares total (SST).

SST : total sum of squares (SST), regression sum of squares (SSR), Sum of square of errors (SSE) are all showing the variation with different measures.

$$SST = \sum_{i=1}^n (y_i - \bar{y})^2$$

$$SSR = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$R^2 = \frac{SSR}{SST} = \frac{\sum (\hat{y}_i - \bar{y})^2}{\sum (y_i - \bar{y})^2}$$

A value of R-squared closer to 1 would mean that the regression model covers most part of the variance of the values of the response variable and can be termed as a good model.

One can alternatively use MSE or R-Squared based on what is appropriate and the need of the hour. However, the disadvantage of using MSE rather than R-squared is that it will be difficult to gauge the performance of the model using MSE as the value of MSE can vary from 0 to any larger number. However, in the case of R-squared, the value is bounded between 0 and 1.

4. Example of Linear Regression

Consider following data for 5 students. Each X_i ($i = 1$ to 5) represents the score of i th student in standard X and corresponding Y_i ($i = 1$ to 5) represents the score of i th student in standard XII.

(i) Linear regression equation best predicts standard XIIth score

(ii) Interpretation for the equation of Linear Regression

(iii) If a student's score is 80 in std X, then what is his expected score in XII standard?

Student	Score in X standard (X_i)	Score in XII standard (Y_i)
1	95	85
2	85	95
3	80	70
4	70	65
5	60	70

x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
95	85	17	8	289	136
85	95	7	18	49	126
80	70	2	-7	4	-14
70	65	-8	-12	64	96

60	70	-18	-7	324	126
$\bar{x} = 78$	$\bar{y} = 77$			$\sum (x - \bar{x})^2 = 730$	$\sum (x - \bar{x})(y - \bar{y}) = 470$

(i) linear regression equation that best predicts standard XIIth score

$$\hat{y} = \beta_0 + \beta_1 x$$

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\beta_1 = 470/730 = 0.644$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_0 = 77 - (0.644 * 78) = 26.768$$

$$\hat{y} = 26.76 + 0.644 x$$

(ii) Interpretation of the regression line. Interpretation 1

For an increase in value of x by 0.644 units there is an increase in value of y in one unit.

Interpretation 2

Even if x = 0 value of independent variable, it is expected that value of y is 26.768 Score in XII standard (Yi) is 0.644 units depending on Score in X standard (Xi) but other factors will also contribute to the result of XII standard by 26.768.

(iii) If a student's score is 65 in std X, then his expected score in XII standard is 78.288

For x = 80 the y value will be $\hat{y} = 26.76 + 0.644 * 65 = 68.38$

5. Training data set and Testing data set

- Machine Learning algorithm has two phases

1. Training and 2. Testing.

- The input of the training phase is training data, which is passed to any machine learning algorithm and machine learning model is generated as output of the training phase.

- The input of the testing phase is test data, which is passed to the machine learning model and prediction is done to observe the correctness of mode.

5. Training data set and Testing data set

- Machine Learning algorithm has two phases

1. Training and 2. Testing.

- The input of the training phase is training data, which is passed to any machine learning algorithm and machine learning model is generated as output of the training phase.
- The input of the testing phase is test data, which is passed to the machine learning model and prediction is done to observe the correctness of mode.

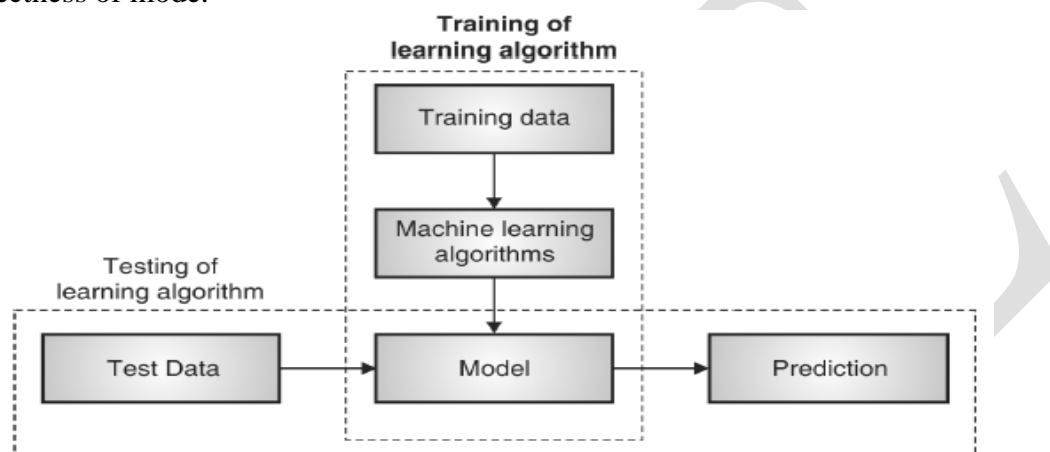


Fig. 1.3.1 : Training and Testing Phase in Machine Learning

(a) Training Phase

- Training dataset is provided as input to this phase.
- Training dataset is a dataset having attributes and class labels and used for training Machine Learning algorithms to prepare models.
- Machines can learn when they observe enough relevant data. Using this one can model algorithms to find relationships, detect patterns, understand complex problems and make decisions.
- Training error is the error that occurs by applying the model to the same data from which the model is trained.
- In a simple way the actual output of training data and predicted output of the model does not match the training error E_{in} is said to have occurred. Training error is much easier to compute.

(b) Testing Phase

- Testing dataset is provided as input to this phase.
- Test dataset is a dataset for which class label is unknown. It is tested using model
- A test dataset used for assessment of the finally chosen model.
- Training and Testing dataset are completely different.
- Testing error is the error that occurs by assessing the model by providing the unknown data to the model.
- In a simple way the actual output of testing data and predicted output of the model does not match the testing error E_{out} is said to have occurred. E_{out} is generally observed larger than E_{in} .

(c) Generalization

- Generalization is the prediction of the future based on the past system.
- It needs to generalize beyond the training data to some future data that it might not have seen yet.
- The ultimate aim of the machine learning model is to minimize the generalization error.
- The generalization error is essentially the average error for data the model has never seen.

- In general, the dataset is divided into two partition training and test sets.
- The fit method is called on the training set to build the model.
- This fit method is applied to the model on the test set to estimate the target value and evaluate the model's performance.
- The reason the data is divided into training and test sets is to use the test set to estimate how well the model trained on the training data and how well it would perform on the unseen data.

Conclusion: In this way we have done data analysis using linear regression for Boston Dataset and predict the price of houses using the features of the Boston Dataset.

Questions:

1) Compute SST, SSE, SSR, MSE, RMSE, R Square for the below example .

Student	Score in X standard (Xi)	Score in XII standard (Yi)
1	95	85
2	85	95
3	80	70
4	70	65
5	60	70

- 2) Comment on whether the model is best fit or not based on the calculated values.
- 3) Write python code to calculate the RSquare for Boston Dataset.
(Consider the linear regression model created in practical session)

Assignment No.	5 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 5

Title: Data Analytics II

1. Implement logistic regression using Python/R to perform classification on Social_Network_Ads.csv dataset.
2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

Prerequisites: 1. Basic of Python Programming
2. Concept of Regression.

Objectives: Students should be able to data analysis using logistic regression using Python for any open source dataset

Theory:

1. Logistic Regression

Classification techniques are an essential part of machine learning and data mining applications. Approximately 70% of problems in Data Science are classification problems. There are lots of classification problems that are available, but logistic regression is common and is a useful regression method for solving the binary classification problem. Another category of classification is Multinomial classification, which handles the issues where multiple classes are present in the target variable. For example, the IRIS dataset is a very famous example of multi-class classification. Other examples are classifying article/blog/document categories.

Logistic Regression can be used for various classification problems such as spam detection. Diabetes prediction, if a given customer will purchase a particular product or will they churn another competitor, whether the user will click on a given advertisement link or not, and many more examples are in the bucket. Logistic Regression is one of the most simple and commonly used Machine Learning algorithms for two-class classification. It is easy to implement and can be used as the baseline for any binary classification problem. Its basic fundamental concepts are also constructive in deep learning. Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables. Logistic regression is a statistical method for predicting binary classes. The outcome or target variable is dichotomous in nature. Dichotomous means there are only two possible classes. For example, it can be used for cancer detection problems. It computes the probability of an event occurring. It is a special case of linear regression where the target variable is categorical in nature. It uses a log of odds as the dependent variable. Logistic Regression predicts the probability of occurrence of a binary event utilising a logit function.

Linear Regression Equation:

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

Where, y is a dependent variable and x1, x2 ... and Xn are explanatory variables.

Sigmoid Function:

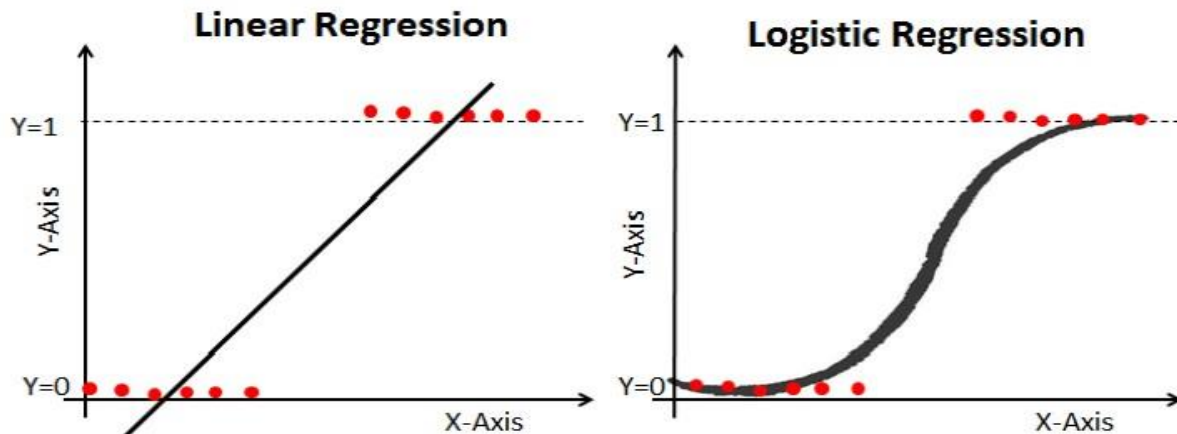
$$p = 1 / (1 + e^{-y})$$

Apply Sigmoid function on linear regression:

$$p = 1 / (1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n)})$$

2. Differentiate between Linear and Logistic Regression

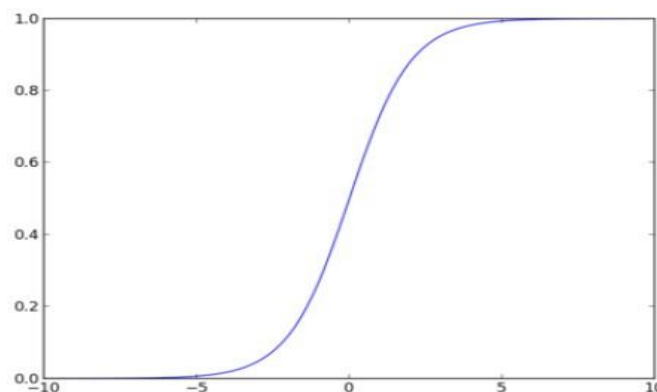
Linear regression gives you a continuous output, but logistic regression provides a constant output. An example of the continuous output is house price and stock price. Examples of the discrete output is predicting whether a patient has cancer or not, predicting whether the customer will churn. Linear regression is estimated using Ordinary Least Squares (OLS) while logistic regression is estimated using Maximum Likelihood Estimation (MLE) approach.



3. Sigmoid Function

The sigmoid function, also called logistic function, gives an 'S' shaped curve that can take any real-valued number and map it into a value between 0 and 1. If the curve goes to positive infinity, y predicted will become 1, and if the curve goes to negative infinity, y predicted will become 0. If the output of the sigmoid function is more than 0.5, we can classify the outcome as 1 or YES, and if it is less than 0.5, we can classify it as 0 or NO. The output cannot be 0 or 1. For example: If the output is 0.75, we can say in terms of probability as: There is a 75 percent chance that a patient will suffer from cancer.

$$f(x) = \frac{1}{1 + e^{-(x)}}$$



4. Types of Logistic Regression

Binary Logistic Regression: The target variable has only two possible outcomes such as Spam or Not Spam, Cancer or No Cancer.

Multinomial Logistic Regression: The target variable has three or more nominal categories such as predicting the type of Wine. **Ordinal Logistic Regression:** the target variable has three or more ordinal categories such as restaurant or product rating from 1 to 5.

5. Confusion Matrix Evaluation Metrics:

Contingency table or Confusion matrix is often used to measure the performance of classifiers. A confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix. The following table shows the confusion matrix for a two class classifier.

		predicted		
		n		
actual	p	TP	FN	P
		FP	TN	N

Confusion matrix

Here each row indicates the actual classes recorded in the test data set and the each column indicates the classes as predicted by the classifier. Numbers on the descending diagonal indicate correct predictions, while the ascending diagonal concerns prediction errors.

Some Important measures derived from confusion matrix are:

- **Number of positive (Pos) :** Total number instances which are labelled as positive in a given dataset.
- **Number of negative (Neg) :** Total number instances which are labelled as negative in a given dataset.
- **Number of True Positive (TP) :** Number of instances which are actually labelled as positive and the predicted class by classifier is also positive.
- **Number of True Negative (TN) :** Number of instances which are actually labelled as negative and the predicted class by classifier is also negative.
- **Number of False Positive (FP) :** Number of instances which are actually labelled as negative and the predicted class by classifier is positive.
- **Number of False Negative (FN):** Number of instances which are actually labelled as positive and the class predicted by the classifier is negative.
- **Accuracy:** Accuracy is calculated as the number of correctly classified instances divided by total number of instances. The ideal value of accuracy is 1, and the worst is 0. It is also calculated as the sum of true positive and true negative (TP + TN) divided by the total number of instances.

$$acc = \frac{TP+TN}{TP+FP+TN+FN} = \frac{TP+TN}{Pos+Neg}$$

● **Error Rate:** Error Rate is calculated as the number of incorrectly classified instances divided by total number of instances. The ideal value of accuracy is 0, and the worst is 1. It is also calculated as the sum of false positive and false negative (FP + FN) divided by the total number of instances.

$$err = \frac{FP+FN}{TP+FP+TN+FN} = \frac{FP+FN}{Pos+Neg} \quad \text{Or}$$

$$err = 1 - acc$$

● **Precision:** It is calculated as the number of correctly classified positive instances divided by the total number of instances which are predicted positive. It is also called confidence value. The ideal value is 1, whereas the worst is 0.

$$precision = \frac{TP}{TP+FP}$$

● **Recall:** .It is calculated as the number of correctly classified positive instances divided by the total number of positive instances. It is also called recall or sensitivity. The ideal value of sensitivity is 1, whereas the worst is 0. It is calculated as the number of correctly classified positive instances divided by the total number of p

$$recall = \frac{TP}{TP+FN}$$

Conclusion: In this way we have done data analysis using logistic regression for Social Media Adv. and evaluate the performance of model.

Value Addition: Visualizing Confusion Matrix using Heat-map

Questions:

1) Consider the binary classification task with two classes positive and negative.

Find out TP, TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall

N = 165	Predicted YES	Predicted NO
Actual YES	TP = 150	FN = 10
Actual NO	FP = 20	TN = 100

2) Comment on whether the model is best fit or not based on the calculated values.

3) Write python code for the preprocessing mentioned in step 4. and Explain every step in detail.

Assignment No.	6 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 6**Title: Data Analytics III**

1. Implement Simple Naïve Bayes classification algorithm using Python/R on iris.csvdataset.
2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

Prerequisites: 1. Basic of Python Programming
2. Concept of Join and Marginal Probability.

Objectives: Students should be able to data analysis using Naive Bayes Algorithm using Python for any open source dataset.

Theory:**1. Concepts used in Naïve Bayes classifier**

- Naïve Bayes Classifier can be used for Classification of categorical data.
- Let there be a 'j' number of classes. $C = \{1, 2, \dots, j\}$
- Let, input observation is specified by 'P' features. Therefore input observation x is given, $x = \{F_1, F_2, \dots, F_p\}$
- The Naïve Bayes classifier depends on Bayes' rule from probability theory.
- Prior probabilities: Probabilities which are calculated for some event based on no other information are called Prior probabilities.

For example, $P(A)$, $P(B)$, $P(C)$ are prior probabilities because while calculating $P(A)$, occurrences of event B or C are not concerned i.e. no information about occurrence of any other event is used.

Conditional Probabilities:

$$P\left(\frac{A}{B}\right) = \frac{P(A \cap B)}{P(B)} \quad \text{if } P(B) \neq 0 \quad \dots \dots (1)$$

$$P\left(\frac{B}{A}\right) = \frac{P(B \cap A)}{P(A)} \quad \dots \dots (2)$$

From equation (1) and (2) ,

$$P(A \cap B) = P\left(\frac{A}{B}\right) \cdot P(B) = P\left(\frac{B}{A}\right) \cdot P(A)$$

$$\therefore P\left(\frac{A}{B}\right) = \frac{P\left(\frac{B}{A}\right) \cdot P(A)}{P(B)}$$

Is called the Bayes Rule.

2. Example of Naive Bayes:

We have a dataset with some features Outlook, Temp, Humidity, and Windy, and the target here is to predict whether a person or team will play tennis or not.

Outlook	Temp	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	TRUE	no

$$X = [\text{Outlook}, \text{Temp}, \text{Humidity}, \text{Windy}]$$

$\underbrace{\hspace{1cm}}_{X_1} \quad \underbrace{\hspace{1cm}}_{X_2} \quad \underbrace{\hspace{1cm}}_{X_3} \quad \underbrace{\hspace{1cm}}_{X_4}$

$$C_k = [\text{Yes}, \text{No}]$$

$\underbrace{\hspace{1cm}}_{C_1} \quad \underbrace{\hspace{1cm}}_{C_2}$

Conditional Probability

$$P(C_k | X) = \frac{P(X | C_k) * P(C_k)}{P(X)}$$

Here, we are predicting the probability of class1 and class2 based on the given condition. If I try to write the same formula in terms of classes and features, we will get the following equation Now we have two classes and four features, so if we write this formula for class C1, it will be something like this.

$$P(C_1 | x_1 \cap x_2 \cap x_3 \cap x_4) = \frac{P(x_1 \cap x_2 \cap x_3 \cap x_4 | C_1) * P(C_1)}{P(x_1 \cap x_2 \cap x_3 \cap x_4)}$$

Here, we replaced Ck with C1 and X with the intersection of X1, X2, X3, X4. You might have a question, It's because we are taking the situation when all these features are present at the same time. The Naive Bayes algorithm assumes that all the features are independent of each other or in other words all the features are unrelated. With that assumption, we can further simplify the above formula and write it in this form

$$P(C_1 | x_1 \cap x_2 \cap x_3 \cap x_4) = \frac{P(x_1 | C_1) * P(x_2 | C_1) * P(x_3 | C_1) * P(x_4 | C_1) * P(C_1)}{P(x_1) * P(x_2) * P(x_3) * P(x_4)}$$

This is the final equation of the Naive Bayes and we have to calculate the probability of both C1 and C2. For this particular example.

Outlook	Temp	Humidity	Windy	Play
Rainy	Cool	High	True	?

$$P(\text{Yes} | X) = P(\text{Rainy} | \text{Yes}) \times P(\text{Cool} | \text{Yes}) \times P(\text{High} | \text{Yes}) \times P(\text{True} | \text{Yes}) \times P(\text{Yes})$$

$$P(\text{Yes} | X) = 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.00529 \rightarrow 0.2 = \frac{0.00529}{0.02057 + 0.00529}$$

$$P(\text{No} | X) = P(\text{Rainy} | \text{No}) \times P(\text{Cool} | \text{No}) \times P(\text{High} | \text{No}) \times P(\text{True} | \text{No}) \times P(\text{No})$$

$$P(\text{No} | X) = 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.02057 \rightarrow 0.8 = \frac{0.02057}{0.02057 + 0.00529}$$

$P(\text{No} | \text{Today}) > P(\text{Yes} | \text{Today})$ So, the prediction that golf would be played is 'No'.

Algorithm (Iris Dataset):

Step 1: Import libraries and create alias for Pandas, Numpy and Matplotlib

Step 2: Import the Iris dataset by calling URL.

Step 3: Initialize the data frame

Step 4: Perform Data Preprocessing

- Convert Categorical to Numerical Values if applicable
- Check for Null Value
- Divide the dataset into Independent(X) and Dependent(Y) variables.
- Split the dataset into training and testing datasets
- Scale the Features if necessary.

Step 5: Use Naive Bayes algorithm(Train the Machine) to Create Model

import the class

from sklearn.naive_bayes import GaussianNB

gaussian = GaussianNB()

gaussian.fit(X_train, y_train)

Step 6: Predict the y_pred for all values of train_x and test_x

Y_pred = gaussian.predict(X_test)

Step 7: Evaluate the performance of Model for train_y and test_y

accuracy = accuracy_score(y_test, Y_pred)

precision = precision_score(y_test, Y_pred, average='micro')

recall = recall_score(y_test, Y_pred, average='micro')

Step 8: Calculate the required evaluation parameters

from sklearn.metrics import

precision_score, confusion_matrix, accuracy_score, recall_score

cm = confusion_matrix(y_test, Y_pred)

Conclusion: In this way we have done data analysis using Naive Bayes Algorithm for Iris dataset and evaluated the performance of the model.

Value Addition: Visualizing Confusion Matrix using Heat-map

Questions:

1) Consider the observation for the car theft scenario having 3 attributes colour, Type and origin.

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Find the probability of car theft having scenarios Red SUV and Domestic.

2) Write python code for the preprocessing mentioned in step 4. and Explain every step in detail.

Assignment No.	7 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 7

Title: Text Analytics:

1. Extract Sample document and apply following document pre-processing methods: Tokenization, POS Tagging, stop words removal, Stemming and Lemmatization.
2. Create representation of document by calculating Term Frequency and Inverse Document Frequency.

Prerequisites:

1. Basic of Python Programming
2. Basic of English language.

Objectives: Students should be able to perform **Text Analysis** using TF IDF Algorithm

Theory:**1. Basic concepts of Text Analytics**

One of the most frequent types of day-to-day conversation is text communication. In our everyday routine, we chat, message, tweet, share status, email, create blogs, and offer opinions and criticism. All of these actions lead to a substantial amount of unstructured text being produced. It is critical to examine huge amounts of data in this sector of the online world and social media to determine people's opinions. Text mining is also referred to as text analytics. Text mining is a process of exploring sizable textual data and finding patterns. Text Mining processes the text itself, while NLP processes with the underlying metadata. Finding frequency counts of words, length of the sentence, presence/absence of specific words is known as text mining. Natural language processing is one of the components of text mining. NLP helps identify sentiment, finding entities in the sentence, and category of blog/article. Text mining is pre-processed data for text analytics. In Text Analytics, statistical and machine learning algorithms are used to classify information.

2. Text Analysis Operations using natural language toolkit:

NLTK(natural language toolkit) is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning and many more. Analysing movie reviews is one of the classic examples to demonstrate a simple NLP Bag-of-words model, on movie reviews.

2.1. Tokenization:

Tokenization is the first step in text analytics. The process of breaking down a text paragraph into smaller chunks such as words or sentences is called Tokenization. Token is a single entity that is the building blocks for a sentence or paragraph.

- Sentence tokenization : split a paragraph into **list of sentences** using **sent_tokenize()** method
- Word tokenization : split a sentence into **list of words** using **word_tokenize()** method

2.2. Stop words removal Stopwords considered as noise in the text. Text may contain stop words such as is, am, are, this, a, an, the, etc. In NLTK for removing stopwords, you need to create a list of stopwords and filter out your list of tokens from these words.

2.3. Stemming and Lemmatization **Stemming:** is a normalization technique where lists of tokenized words are converted into shortened root words to remove redundancy. Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form. A computer program that stems word may be called a stemmer. E.g. A stemmer reduces the words like fishing, fished, and fisher to the stem fish. The stem need

not be a word, for example the Porter algorithm reduces, argue, argued, argues, arguing, and argus to the stem argument

Lemmatization in NLTK is the algorithmic process of finding the lemma of a word depending on its meaning and context. Lemmatization usually refers to the morphological analysis of words, which aims to remove inflectional endings. It helps in returning the base or dictionary form of a word known as the lemma. Eg. Lemma for studies is study

1) Lemmatization Vs Stemming

Stemming algorithm works by cutting the suffix from the word. In a broader sense cuts either the beginning or end of the word. On the contrary, Lemmatization is a more powerful operation, and it takes into consideration morphological analysis of the words. It returns the lemma which is the base form of all its inflectional forms. In-depth linguistic knowledge is required to create dictionaries and look for the proper form of the word.

Stemming is a general operation while lemmatization is an intelligent operation where the proper form will be looked in the dictionary. Hence, lemmatization helps in forming better machine learning features.

2.4. POS Tagging

POS (Parts of Speech) tell us about grammatical information of words of the sentence by assigning specific token (Determiner, noun, adjective, adverb, verb, Personal Pronoun etc.) as tag (DT, NN, JJ, RB, VB, PRP etc) to each words. Word can have more than one POS depending upon the context where it is used. We can use POS tags as statistical NLP tasks. It distinguishes a sense of word which is very helpful in text realization and infer semantic information from text for sentiment analysis.

3. Text Analysis Model using TF-IDF.

Term frequency-inverse document frequency (TFIDF), is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

• Term Frequency (TF)

It is a measure of the frequency of a word (w) in a document (d). TF is defined as the ratio of a word's occurrence in a document to the total number of words in a document. The denominator term in the formula is to normalize since all the corpus documents are of different lengths.

$$TF(w, d) = \frac{\text{occurrences of } w \text{ in document } d}{\text{total number of words in document } d}$$

Example:

Documents	Text	Total number of words in a document
A	Jupiter is the largest planet	5
B	Mars is the fourth planet from the sun	8

The initial step is to make a vocabulary of unique words and calculate TF for each document. TF will be more for words that frequently appear in a document and less for rare words in a document.

• Inverse Document Frequency (IDF)

It is the measure of the importance of a word. Term frequency (TF) does not consider the importance of words. Some words such as 'of', 'and', etc. can be most frequently present but are of little significance. IDF provides weightage to each word based on its frequency in the corpus D.

$$IDF(w, D) = \ln\left(\frac{\text{Total number of documents (N) in corpus D}}{\text{number of documents containing } w}\right)$$

In our example, since we have two documents in the corpus, N=2.

Words	TF (for A)	TF (for B)	IDF
Jupiter	1/5	0	$\ln(2/1) = 0.69$
Is	1/5	1/8	$\ln(2/2) = 0$
The	1/5	2/8	$\ln(2/2) = 0$
largest	1/5	0	$\ln(2/1) = 0.69$
Planet	1/5	1/8	$\ln(2/2) = 0$
Mars	0	1/8	$\ln(2/1) = 0.69$
Fourth	0	1/8	$\ln(2/1) = 0.69$
From	0	1/8	$\ln(2/1) = 0.69$
Sun	0	1/8	$\ln(2/1) = 0.69$

Term Frequency — Inverse Document Frequency (TFIDF)

It is the product of TF and IDF. TFIDF gives more weightage to the word that is rare in the corpus (all the documents). TFIDF provides more importance to the word that is more frequent in the document.

$$TFIDF(w, d, D) = TF(w, d) * IDF(w, D)$$

Words	TF (for A)	TF (for B)	IDF	TFIDF (A)	TFIDF (B)
Jupiter	1/5	0	$\ln(2/1) = 0.69$	0.138	0
Is	1/5	1/8	$\ln(2/2) = 0$	0	0
The	1/5	2/8	$\ln(2/2) = 0$	0	0
largest	1/5	0	$\ln(2/1) = 0.69$	0.138	0
Planet	1/5	1/8	$\ln(2/2) = 0$	0.138	0
Mars	0	1/8	$\ln(2/1) = 0.69$	0	0.086
Fourth	0	1/8	$\ln(2/1) = 0.69$	0	0.086
From	0	1/8	$\ln(2/1) = 0.69$	0	0.086
Sun	0	1/8	$\ln(2/1) = 0.69$	0	0.086

After applying TFIDF, text in A and B documents can be represented as a TFIDF vector of dimension equal to the vocabulary words. The value corresponding to each word represents the importance of that word in a particular document. TFIDF is the product of TF with IDF. Since TF values lie between 0 and 1, not using \ln can result in high IDF for some words, thereby dominating the TFIDF. We don't want that, and therefore, we use \ln so that the IDF should not completely dominate the TFIDF.

- **Disadvantage of TFIDF**

It is unable to capture the semantics. For example, funny and humorous are synonyms, but TFIDF does not capture that. Moreover, TFIDF can be computationally expensive if the vocabulary is vast.

4. Bag of Words (BoW)

Machine learning algorithms cannot work with raw text directly. Rather, the text must be converted into vectors of numbers. In natural language processing, a common technique for extracting features from text is to place all of the words that occur in the text in a bucket. This approach is called a bag of words model or BoW for short. It's referred to as a "bag" of words because any information about the structure of the sentence is lost.

Conclusion: In this way we have done text data analysis using TF IDF algorithm

Questions:

Q.1) Perform Stemming for text = "*studies studying cries cry*". Compare the results generated with Lemmatization. Comment on your answer how Stemming and Lemmatization differ from each other.

Q.2) Write Python code for removing stop words from the below documents, convert the documents into lowercase and calculate the TF, IDF and TFIDF score for each document.

documentA = '*Jupiter is the largest Planet*'

documentB = '*Mars is the fourth planet from the Sun*'

Assignment No.	8 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 8

Title: Data Visualization I

1. Use the inbuilt dataset 'titanic'. The dataset contains 891 rows and contains information about the passengers who boarded the unfortunate Titanic ship. Use the Seaborn library to see if we can find any patterns in the data.
2. Write a code to check how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram.

Prerequisites:

1. Basic of Python Programming
2. Seaborn Library, Concept of Data Visualization

Objectives: Student should be able to perform the data visualization operation using Python on any open source dataset.

Theory:

Data Visualisation plays a very important role in Data mining. Various data scientists spent their time exploring data through visualisation. To accelerate this process we need to have a well-documentation of all the plots. Even plenty of resources can't be transformed into valuable goods without planning and architecture

1. Seaborn Library Basics

Seaborn is a Python data visualisation library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. For the installation of Seaborn, you may run any of the following in your command line.

pip install seaborn

conda install seaborn

To import seaborn you can run the following command.

import seaborn as sns

2. Know your data

The dataset that we are going to use to draw our plots will be the Titanic dataset, which is downloaded by default with the Seaborn library. All you have to do is use the `load_dataset` function and pass it the name of the dataset.

`dataset.head()`

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

The dataset contains 891 rows and 15 columns and contains information about the passengers who boarded the unfortunate Titanic ship. The original task is to predict whether or not the passenger survived depending upon different features such as their age, ticket, cabin they boarded, the class of the ticket, etc. We will use the Seaborn library to see if we can find any patterns in the data.

3. Finding patterns of data. Patterns of data can be find out with the help of different types of plots

Types of plots are:

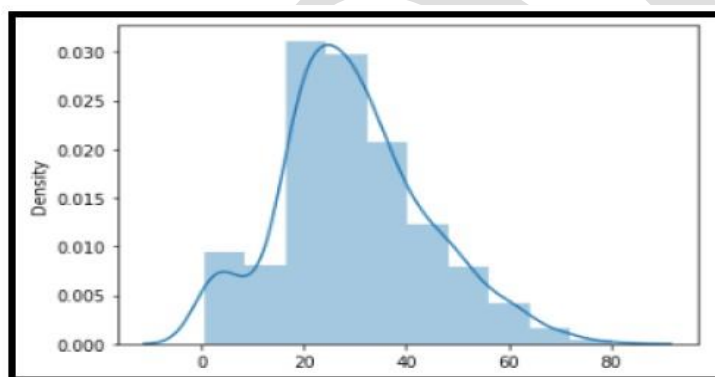
A. Distribution Plots	B. Categorical Plots	C. Advanced Plots	D. Matrix Plots
a. Dist-Plot	a. Bar Plot	a. Strip Plot	a. Heat Map
b. Joint Plot	b. Count Plot	b. Swarm Plot	b. Cluster Map
c. Rug Plot	c. Box Plot		
	d. Violin Plot		

A. Distribution Plots:

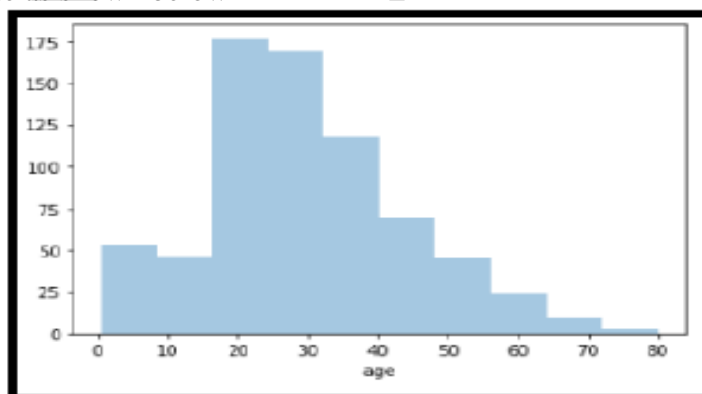
These plots help us to visualise the distribution of data. We can use these plots to understand the mean, median, range, variance, deviation, etc of the data.

a. Distplot

Dist plot gives us the histogram of the selected continuous variable. It is an example of a univariate analysis. We can change the number of bins i.e. number of vertical bars in a histogram



The line that you see represents the kernel density estimation. You can remove this line by passing False as the parameter for the kde attribute as shown below

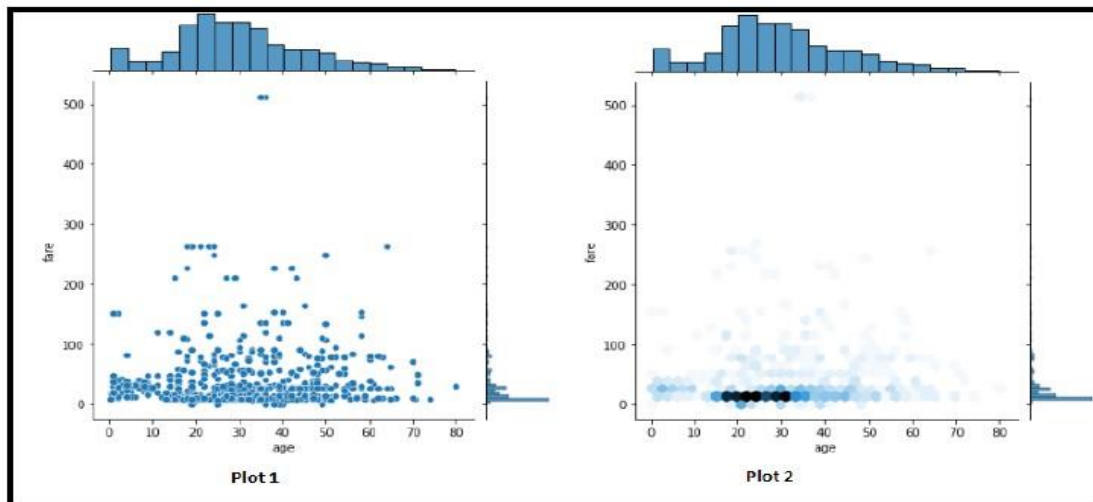


Here the x-axis is the age and the y-axis displays frequency. For example, for bins = 10, there are around 50 people having age 0 to 10

i.b. Joint Plot

It is the combination of the distplot of two variables. It is an example of bivariate analysis. We additionally obtain a scatter plot between the variables to reflect their linear relationship. We can customize the scatter plot into a hexagonal plot, where, the more the colour intensity, the more will be the number of observations.

```
import seaborn as sns
# For Plot 1
sns.jointplot(x = dataset['age'], y = dataset['fare'], kind = 'scatter')
# For Plot 2
sns.jointplot(x = dataset['age'], y = dataset['fare'], kind = 'hex')
```

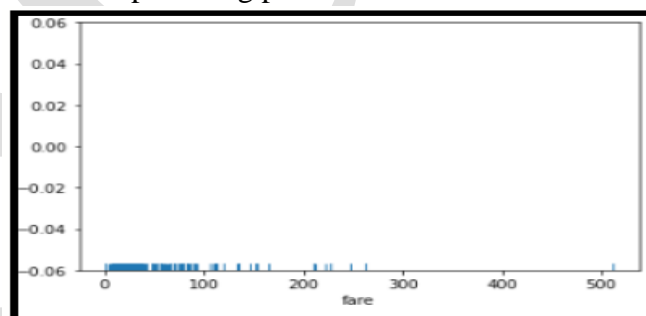


From the output, you can see that a joint plot has three parts. A distribution plot at the top for the column on the x-axis, a distribution plot on the right for the column on the y-axis and a scatter plot in between that shows the mutual distribution of data for both the columns. You can see that there is no correlation observed between prices and the fares. You can change the type of the joint plot by passing a value for the kind parameter. For instance, if instead of a scatter plot, you want to display the distribution of data in the form of a hexagonal plot, you can pass the value hex for the kind parameter.

c. The Rug Plot

b. The rugplot() is used to draw small bars along the x-axis for each point in the dataset. To plot a rug plot, you need to pass the name of the column. Let's plot a rug plot for fare.

```
sns.rugplot(dataset['fare'])
```



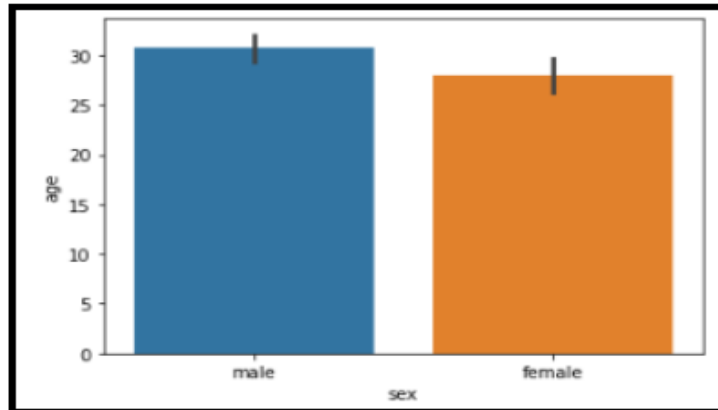
From the output, you can see that most of the instances for the fares have values between 0 and 100. These are some of the most commonly used distribution plots offered by the Python's Seaborn Library. Let's see some of the categorical plots in the Seaborn library.

2. Categorical Plots

Categorical plots, as the name suggests, are normally used to plot categorical data. The categorical plots plot the values in the categorical column against another categorical column or a numeric column. Let's see some of the most commonly used categorical data.

b. The Bar Plot: The `barplot()` is used to display the mean value for each value in a categorical column, against a numeric column. The first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset. For instance, if you want to know the mean value of the age of the male and female passengers, you can use the bar plot as follows.

```
sns.barplot(x='sex', y='age', data=dataset)
```

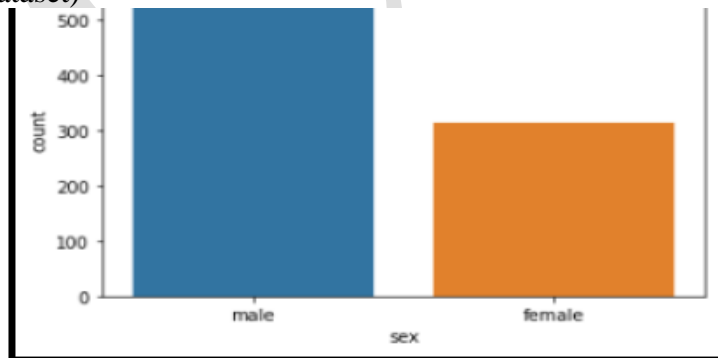


From the output, you can clearly see that the average age of male passengers is just less than 40 while the average age of female passengers is around 33. In addition to finding the average, the bar plot can also be used to calculate other aggregate values for each category. To do so, you need to pass the aggregate function to the estimator.

c. The Count Plot

The count plot is similar to the bar plot, however it displays the count of the categories in a specific column. For instance, if we want to count the number of males and women passenger we can do so using count plot as follows:

```
sns.countplot(x='sex', data=dataset)
```

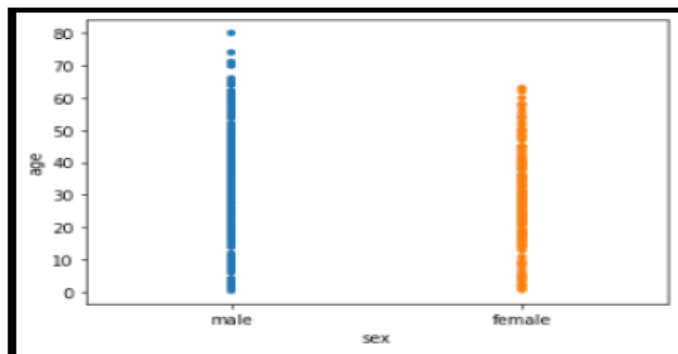


Advanced Plots:

a. The Strip Plot

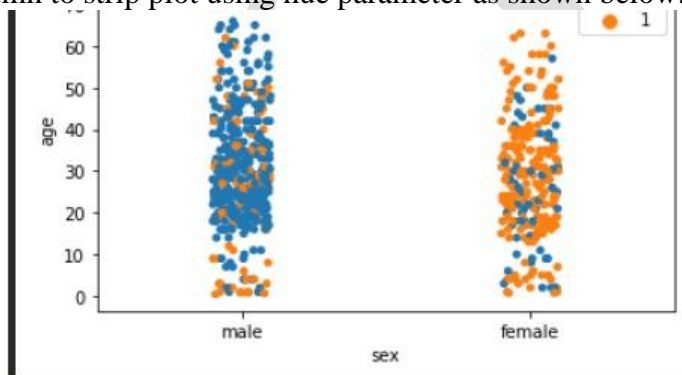
The strip plot draws a scatter plot where one of the variables is categorical. We have seen scatter plots in the joint plot and the pair plot sections where we had two numeric variables. The strip plot is different in a way that one of the variables is categorical in this case, and for each category in the categorical variable, you will see a scatter plot with respect to the numeric column. Look at the following script:

```
sns.stripplot(x='sex', y='age', data=dataset, jitter=False)
```



You can see the scattered plots of age for both males and females. The data points look like strips. It is difficult to comprehend the distribution of data in this form. To better comprehend the data, pass True for the jitter parameter which adds some random noise to the data.

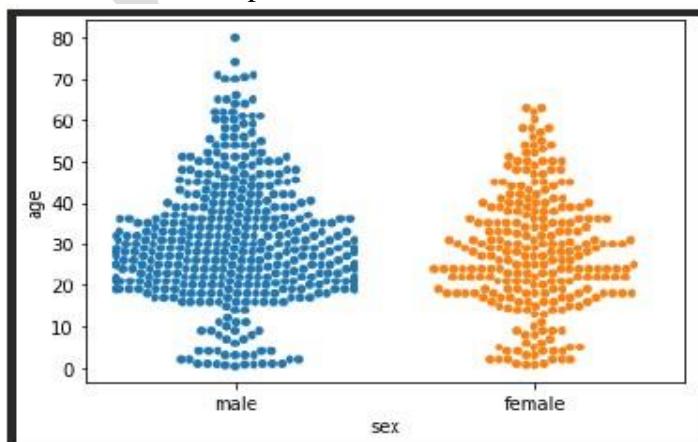
Now you have a better view for the distribution of age across the genders. Like violin and box plots, you can add an additional categorical column to strip plot using hue parameter as shown below:



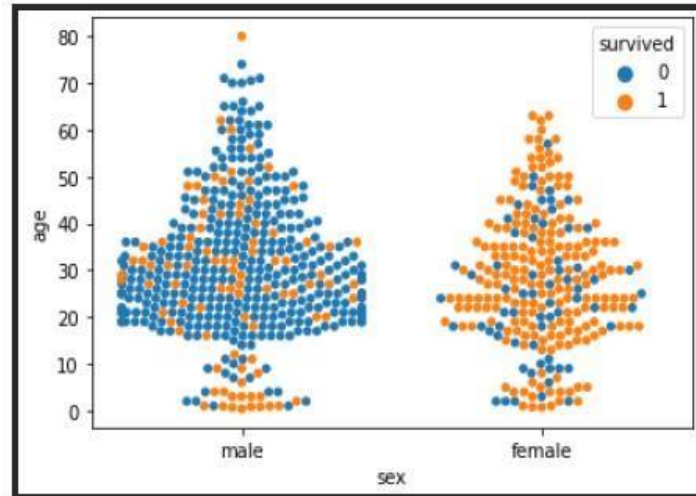
b. The Swarm Plot

The swarm plot is a combination of the strip and the violin plots. In the swarm plots, the points are adjusted in such a way that they don't overlap. Let's plot a swarm plot for the distribution of age against gender. The `swarmplot()` function is used to plot the violin plot. Like the box plot, the first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset.

Example:



You can clearly see that the above plot contains scattered data points like the strip plot and the data points are not overlapping. Rather they are arranged to give a view similar to that of a violin plot. Let's add another categorical column to the swarm plot using the hue parameter.

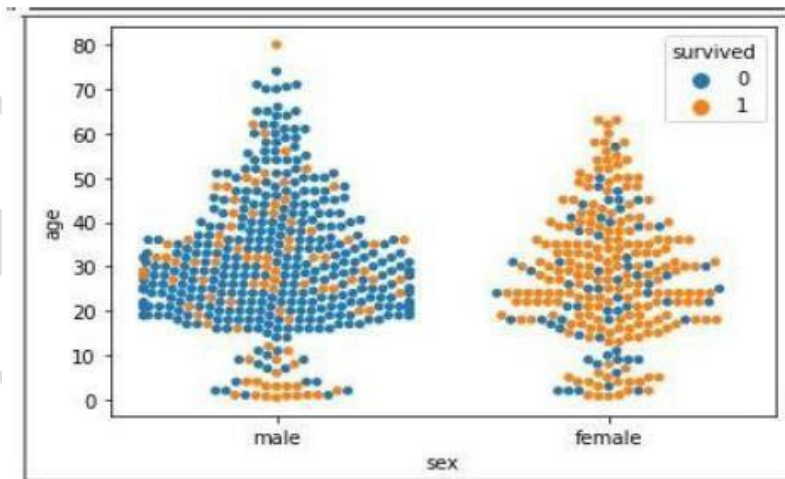


Conclusion:

Seaborn is an advanced data visualisation library built on top of Matplotlib library. In this assignment, we looked at how we can draw distributional and categorical plots using the Seaborn library. We have seen how to plot matrix plots in Seaborn. We also saw how to change plot styles and use grid functions to manipulate subplots.

Questions:

1. List out different types of plot to find patterns of data
2. Explain when you will use distribution plots and when you will use categorical plots.
3. Write the conclusion from the following swarm plot (consider titanic dataset)



4. Which parameter is used to add another categorical variable to the violin plot, Explain with syntax and example

Assignment No.	9 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 9

Title: Data Visualization II

1. Use the inbuilt dataset 'titanic' as used in the above problem. Plot a box plot for distribution of age with respect to each gender along with the information about whether they survived or not. (Column names : 'sex' and 'age') Write observations on the inference from the above statistics.

Prerequisites:

1. Basic of Python Programming
2. Seaborn Library, Concept of Data Visualization

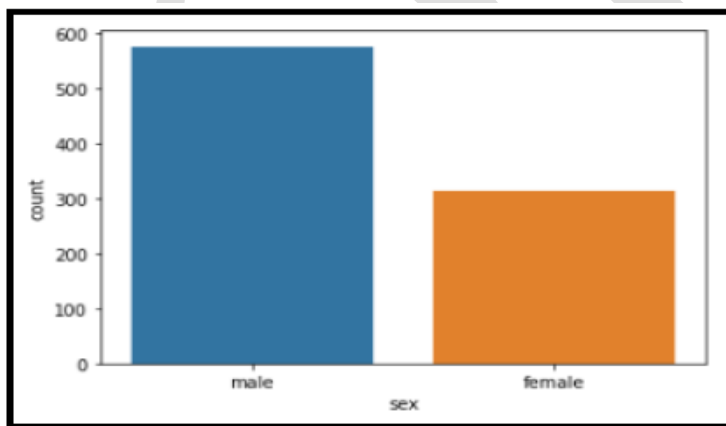
Objectives: Student should be able to perform the data visualization operation using Python on any open source dataset.

Theory:

The Count Plot

The count plot is similar to the bar plot, however it displays the count of the categories in a specific column. For instance, if we want to count the number of males and women passenger we can do so using count plot as follows:

```
sns.countplot(x='sex', data=dataset)
```



Pie Plot:

A **Pie Chart** is a circular statistical plot that can display only one series of data. The area of the chart is the total percentage of the given data. The area of slices of the pie represents the percentage of the parts of the data. The slices of pie are called wedges. The area of the wedge is determined by the length of the arc of the wedge.

Creating Pie Plot:

Matplotlib API has `pie()` function in its `pyplot` module which create a pie chart representing the data in an array.

Syntax: `matplotlib.pyplot.pie(data, explode=None, labels=None, colors=None, autopct=None, shadow=False)`

Parameters:

data represents the array of data values to be plotted, the fractional area of each slice is represented by **data/sum(data)**. If `sum(data)<1`, then the data values returns the fractional area directly, thus resulting pie will have empty wedge of size `1-sum(data)`.

labels is a list of sequence of strings which sets the label of each wedge.

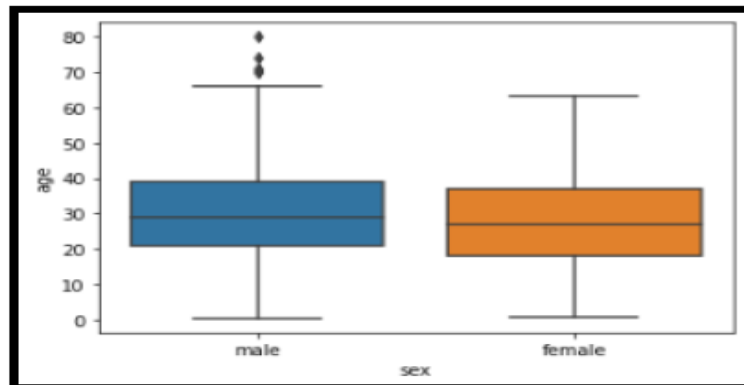
color attribute is used to provide color to the wedges.

autopct is a string used to label the wedge with their numerical value.

shadow is used to create shadow of wedge.

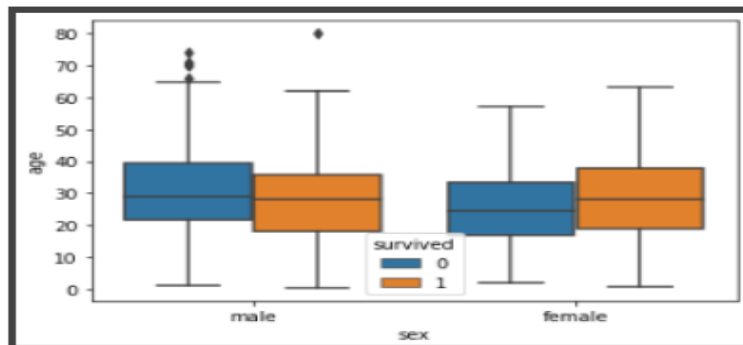
The Box Plot:

The box plot is used to display the distribution of the categorical data in the form of quartiles. The centre of the box shows the median value. The value from the lower whisker to the bottom of the box shows the first quartile. From the bottom of the box to the middle of the box lies the second quartile. From the middle of the box to the top of the box lies the third quartile and finally from the top of the box to the top whisker lies the last quartile. Now let's plot a box plot that displays the distribution for the age with respect to each gender.



Let's try to understand the box plot for females. The first quartile starts at around 1 and ends at 20 which means that 25% of the passengers are aged between 1 and 20. The second quartile starts at around 20 and ends at around 28 which means that 25% of the passengers are aged between 20 and 28. Similarly, the third quartile starts and ends between 28 and 38, hence 25% passengers are aged within this range and finally the fourth or last quartile starts at 38 and ends around 64.

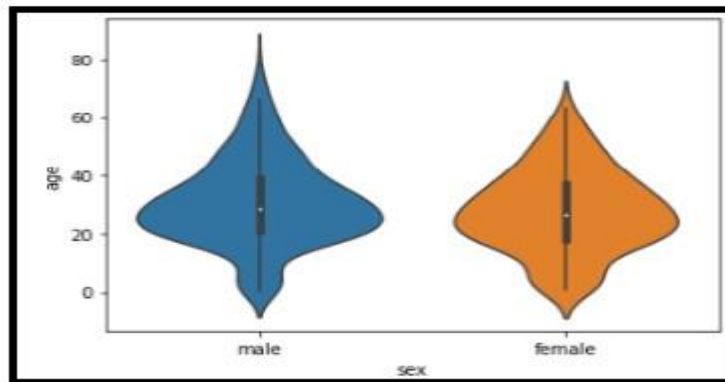
You can make your box plots more fancy by adding another layer of distribution. For instance, if you want to see the box plots of forage of passengers of both genders, along with the information about whether or not they survived, you can pass the survived as value to the hue parameter as shown below:



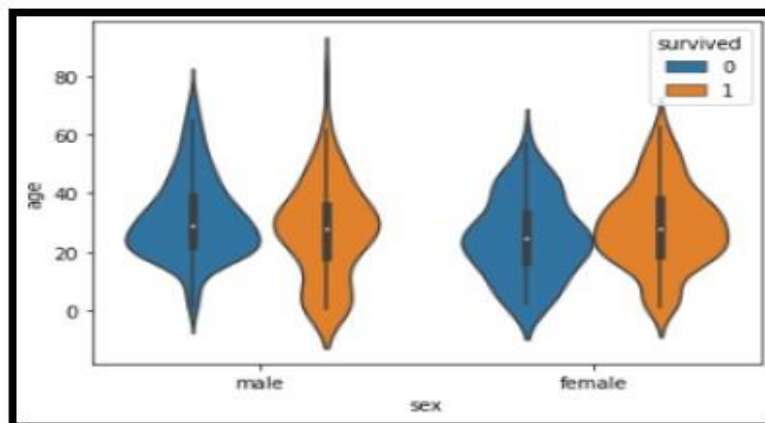
The Violin Plot

The violin plot is similar to the box plot, however, the violin plot allows us to display all the components that actually correspond to the data point. The violin plot() function is used to plot the violin plot. Like the box plot, the first parameter is the categorical column, the second parameter is the numeric column while the third parameter is the dataset. Let's plot a violin plot that displays the distribution for the age with respect to each gender.

```
sns.violinplot(x='sex', y='age', data=dataset)
```



You can see from the figure above that violin plots provide much more information about the data as compared to the box plot. Instead of plotting the quartile, the violin plot allows us to see all the components that actually correspond to the data. The area where the violin plot is thicker has a higher number of instances for the age. For instance, from the violin plot for males, it is clearly evident that the number of passengers with age between 20 and 40 is higher than all the rest of the age brackets. Like box plots, you can also add another categorical variable to the violin plot using the hue parameter as shown below:



Matrix Plots

Matrix plots are the type of plots that show data in the form of rows and columns. Heat maps are the prime examples of matrix plots.

a. Heat Maps

Heat maps are normally used to plot correlation between numeric columns in the form of a matrix. It is important to mention here that to draw matrix plots, you need to have meaningful information on rows as well as columns.

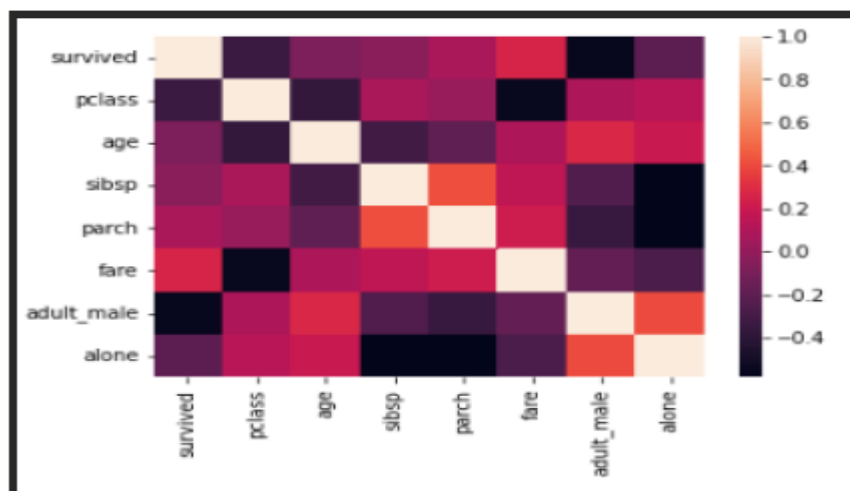
The `corr()` function returns the correlation between all the numeric columns of the dataset. Execute the following script: `dataset.corr()`

In the output, you will see that both the columns and the rows have meaningful header information, as shown below:

	survived	pclass	age	sibsp	parch	fare	adult_male	alone
survived	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307	-0.557080	-0.203367
pclass	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500	0.094035	0.135207
age	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067	0.280328	0.198270
sibsp	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651	-0.253586	-0.584471
parch	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225	-0.349943	-0.583398
fare	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000	-0.182024	-0.271832
adult_male	-0.557080	0.094035	0.280328	-0.253586	-0.349943	-0.182024	1.000000	0.404744
alone	-0.203367	0.135207	0.198270	-0.584471	-0.583398	-0.271832	0.404744	1.000000

Now to create a heat map with these correlation values, you need to call the heatmap() function and pass it your correlation dataframe. Look at the following script:

```
corr = dataset.corr()
sns.heatmap(corr)
```



b. Cluster Map:

In addition to the heat map, another commonly used matrix plot is the cluster map. The cluster map basically uses Hierarchical Clustering to cluster the rows and columns of the matrix.

Conclusion: In this way we have done how we can draw Boxplot and heatmap using the Seaborn library. We have seen how to plot matrix plots in Seaborn.

Questions:

1. Write down the code to use inbuilt dataset 'titanic' using seaborn library.
2. Write code to plot a box plot for distribution of age with respect to each gender along with the information about whether they survived or not.
3. Write the observations from the box plot.

Assignment No.	10 (GROUP A)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 10

Title: Data Visualization III

Download the Iris flower dataset or any other dataset into a DataFrame.

(e.g., <https://archive.ics.uci.edu/ml/datasets/Iris>). Scan the dataset and give the inference as:

1. List down the features and their types (e.g., numeric, nominal) available in the dataset.
2. Create a histogram for each feature in the dataset to illustrate the feature distributions.
3. Create a box plot for each feature in the dataset.
4. Compare distributions and identify outliers.

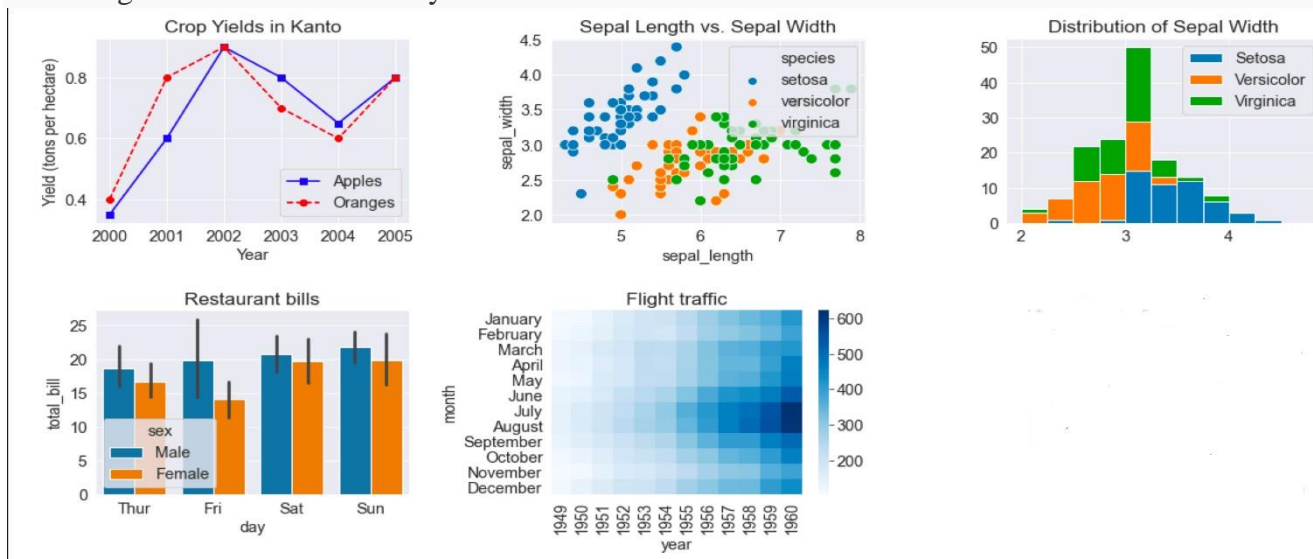
Prerequisites:

1. Basic of Python Programming
2. Seaborn Library, Concept of Data Visualization.
3. Types of variables

Objectives: Students should be able to perform the data Visualization operation using Python on any open source dataset

Theory:

Data Visualization: Data visualization is the graphical representation of information and data. By using [visual elements like charts, graphs, and maps](#), data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. Additionally, it provides an excellent way for employees or business owners to present data to non-technical audiences without confusion. In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.



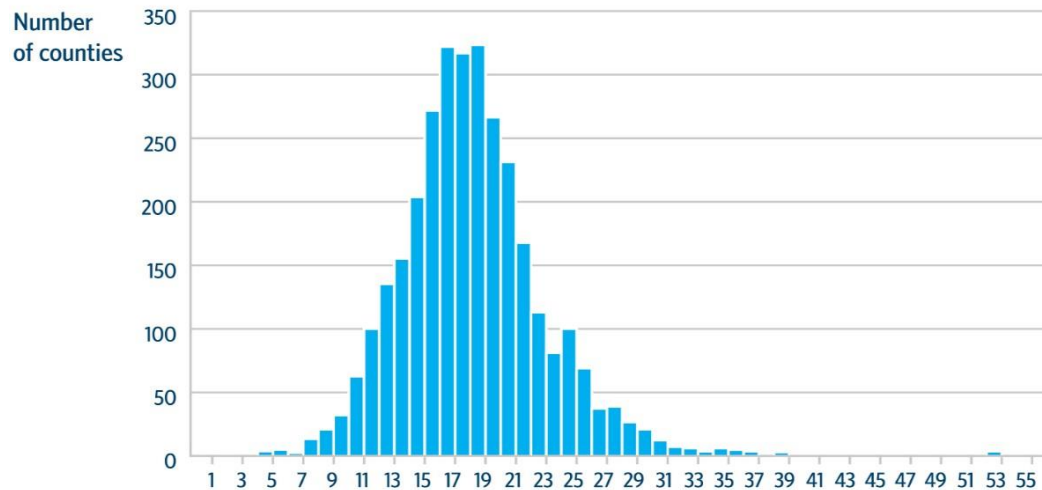
Histogram in Data Visualization:

A histogram is a chart that displays numeric data in ranges, where each bar represents how frequently numbers fall into a particular range.

Like a bar chart, histograms consist of a series of vertical bars along the x-axis. Histograms are most commonly used to depict what a set of data looks like in aggregate. At a quick glance, histograms tell whether a dataset has values that are clustered around a small number of ranges or are more spread out.

Example:

Item 1: Percentage of Population Aged 65 and Older



Percentage of seniors (aged 65 years and over) in each county

Boxplot in Data Visualization:

Box Plot is the visual representation of the depicting groups of numerical data through their quartiles. Boxplot is also used for detect the outlier in data set. It captures the summary of the data efficiently with a simple box and whiskers and allows us to compare easily across groups. Boxplot summarizes a sample data using 25th, 50th and 75th percentiles. These percentiles are also known as the lower quartile, median and upper quartile.

A box plot consist of 5 things.

- Minimum
- First Quartile or 25%
- Median (Second Quartile) or 50%
- Third Quartile or 75%
- Maximum

- **Draw the boxplot using seaborn library:**

- **Syntax :**

```
seaborn.boxplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None,
orient=None, color=None, palette=None, saturation=0.75, width=0.8, dodge=True, fliersize=5,
linewidth=None, whis=1.5, notch=False, ax=None, **kwargs)
```

- **Parameters:**

x = feature of dataset

y = feature of dataset

hue = feature of dataset

data = dataframe or full dataset

color = color name

Example:

```
# load the dataset
```

```
tips = sns.load_dataset('tips')
```

```
tips.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

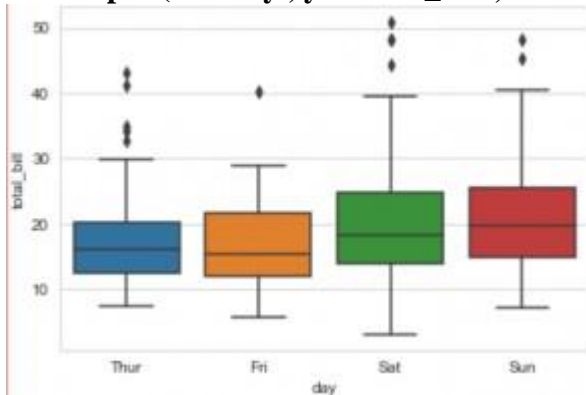
Boxplot of days with respect total_bill.

Draw a vertical boxplot grouped

by a categorical variable:

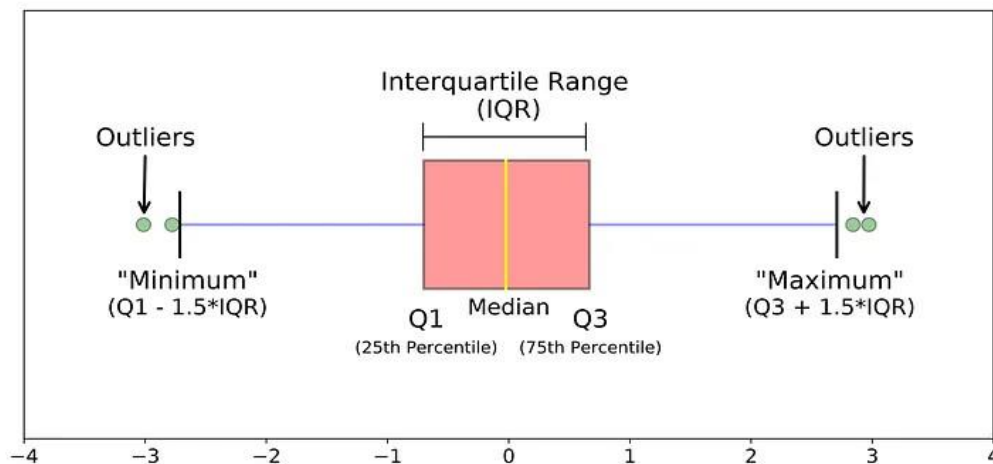
`sns.set_style("whitegrid")`

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



Box Plot Visualize Outlier:

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In a sense, this definition leaves it up to the analyst (or a consensus process) to decide what will be considered abnormal. outlier visualization the box plot is the easiest way to grasp valuable information about your data's outliers. But before visualizing any outliers let's understand what's a box plot and its different components:



As we can see in the image above, a box plot has a lot of components and every one of them helps us to represent and understand the data:

- **Q1.** 25% of the data is below this data point.
- **Median.** The central value of the data set. It can also be represented as Q2. 50% of the data is below this data point.
- **Q3.** 75% of the data is below this data point.
- **Minimum.** The data point with the smallest value in the data set that isn't an outlier.
- **Maximum.** The data point with the biggest value in the data set that isn't an outlier.
- **IQR.** Represents all the values between Q1 and Q3.

Once we understood all the components of a box plot let's visualize it for a given variable in our data set:

```
fig = plt.figure(figsize=(10,5))
```

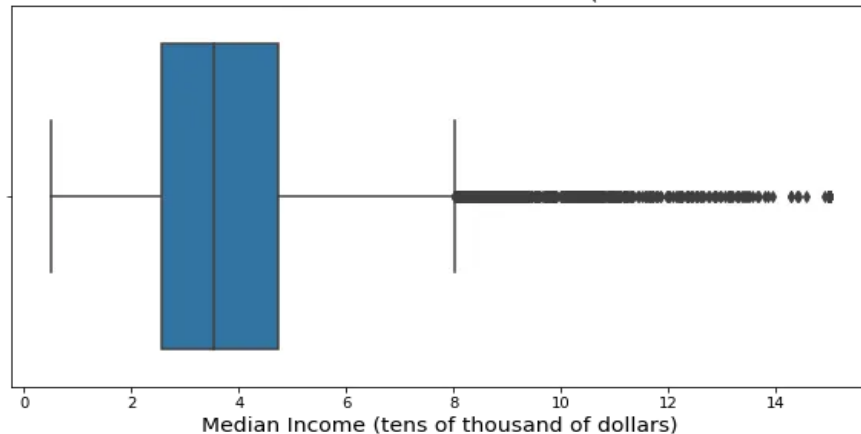
```
sns.boxplot(df.MedInc)
```

```
plt.title('Box Plot: Median income for households within a block (tens of thousands of dollars)', fontsize=15)
```

```
plt.xlabel('Median Income (tens of thousand of dollars)', fontsize=14)
```

```
plt.show()
```

Box Plot: Median income for households within a block (tens of thousands of dollars)



Conclusion:

In this way we have done how we can draw Boxplot and compare distributions and identify outliers using the Seaborn library. We have seen how to histogram in Seaborn.

Questions:

1. For the iris dataset, list down the features and their types.
2. Write a code to create a histogram for each feature. (iris dataset)
3. Write a code to create a boxplot for each feature. (iris dataset)
4. Identify the outliers from the boxplot drawn for iris dataset.

GROUP B: BIG DATA ANALYTICS

Assignment No.	1 (GROUP B)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 1

Title: Write a code in JAVA for a simple Word Count application that counts the number of occurrences of each word in a given input set using the Hadoop Map-Reduce framework on local-standalone set-up

Prerequisites:

- **Java Installation** - Check whether the Java is installed or not using the following command.
`java -version`
- **Hadoop Installation** - Check whether the Hadoop is installed or not using the following command.
`hadoop version`

Objectives: Students should be able to perform the Word count application using Java on Hadoop Map reduce framework.

Theory:

Hadoop:

Apache Hadoop is an open source framework that is used to efficiently store and process large datasets ranging in size from gigabytes to petabytes of data. Instead of using one large computer to store and process the data, Hadoop allows clustering multiple computers to analyze massive datasets in parallel more quickly.

Hadoop consists of four main modules:

- Hadoop Distributed File System (HDFS) – A distributed file system that runs on standard or low-end hardware. HDFS provides better data throughput than traditional file systems, in addition to high fault tolerance and native support of large datasets.
- Yet Another Resource Negotiator (YARN) – Manages and monitors cluster nodes and resource usage. It schedules jobs and tasks.
- MapReduce – A framework that helps programs do the parallel computation on data. The map task takes input data and converts it into a dataset that can be computed in key value pairs. The output of the map task is consumed by reduce tasks to aggregate output and provide the desired result.
- Hadoop Common – Provides common Java libraries that can be used across all modules.

MapReduce?

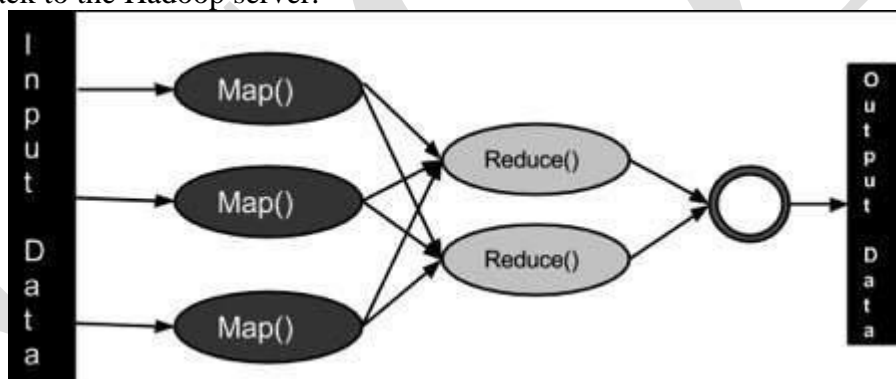
MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job.

The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into *mappers* and *reducers* is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines

in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model.

The Algorithm

- Generally MapReduce paradigm is based on sending the computer to where the data resides!
- MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage.
 - **Map stage** – The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
 - **Reduce stage** – This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.
- During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.
- The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.
- Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
- After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.



Inputs and Outputs (Java Perspective)

The MapReduce framework operates on <key, value> pairs, that is, the framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job, conceivably of different types.

The key and the value classes should be in serialized manner by the framework and hence, need to implement the Writable interface. Additionally, the key classes have to implement the Writable-Comparable interface to facilitate sorting by the framework. Input and Output types of a **MapReduce job** – (Input) <k1, v1> → map → <k2, v2> → reduce → <k3, v3>(Output).

	Input	Output
Map	$\langle k1, v1 \rangle$	list ($\langle k2, v2 \rangle$)
Reduce	$\langle k2, \text{list}(v2) \rangle$	list ($\langle k3, v3 \rangle$)

In Hadoop, [MapReduce](#) is a computation that decomposes large manipulation jobs into individual tasks that can be executed in parallel across a cluster of servers. The results of tasks can be joined together to compute final results.

MapReduce consists of 2 steps:

- **Map Function** – It takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (Key-Value pair).

Example – (Map function in Word Count)

Input	Set of data	Bus, Car, bus, car, train, car, bus, car, train, bus, TRAIN, BUS, buS, caR, CAR, car, BUS, TRAIN
Output	Convert into another set of data (Key, Value)	(Bus,1), (Car,1), (bus,1), (car,1), (train,1), (car,1), (bus,1), (car,1), (train,1), (bus,1), (TRAIN,1), (BUS,1), (buS,1), (caR,1), (CAR,1), (car,1), (BUS,1), (TRAIN,1)

- **Reduce Function** – Takes the output from Map as an input and combines those data tuples into a smaller set of tuples.

Example – (Reduce function in Word Count)

Input (output of Map function)	Set of Tuples	(Bus,1), (Car,1), (bus,1), (car,1), (train,1), (car,1), (bus,1), (car,1), (train,1), (bus,1), (TRAIN,1),(BUS,1), (buS,1), (caR,1), (CAR,1), (car,1), (BUS,1), (TRAIN,1)
Output	Converts into smaller set of tuples	(BUS,7), (CAR,7), (TRAIN,4)

Work Flow of the Program

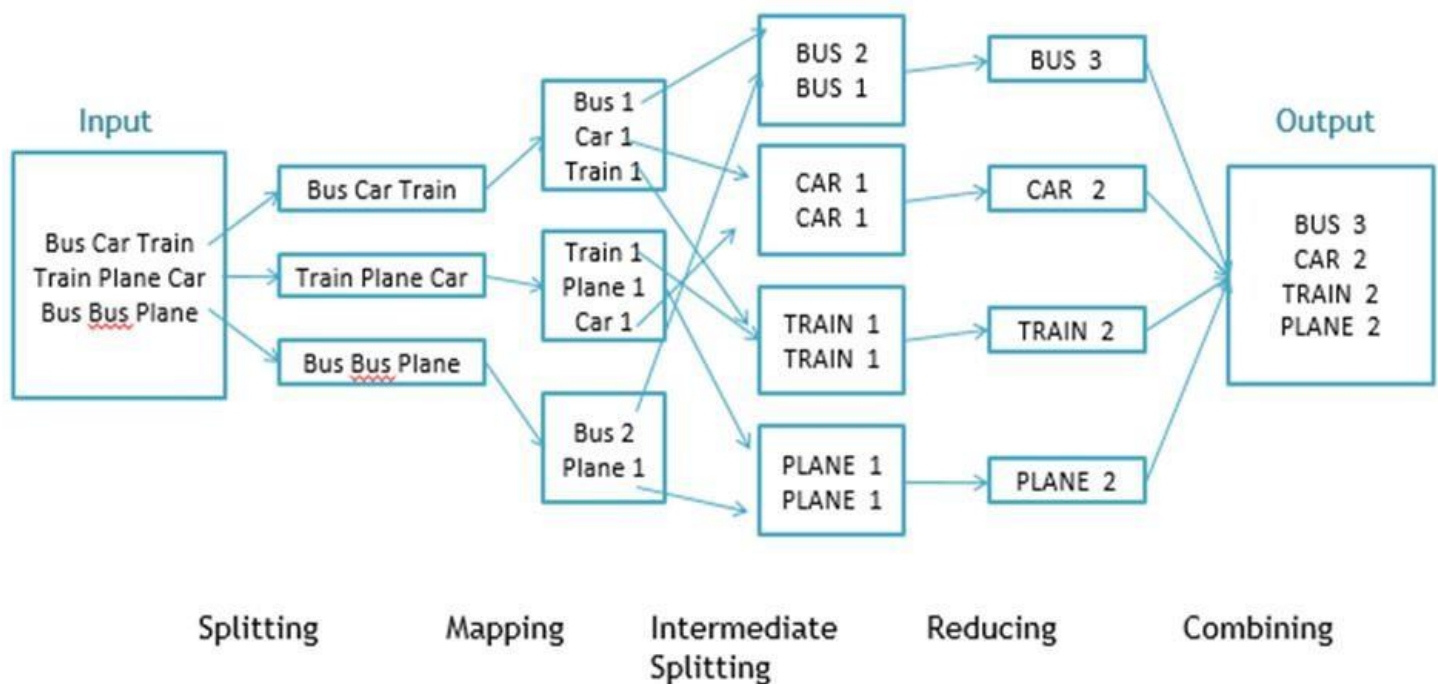


Fig. WorkFlow of MapReducing

Workflow of MapReduce consists of 5 steps:

1. **Splitting** – The splitting parameter can be anything, e.g. splitting by space, comma, semicolon, or even by a new line ('\n').
2. **Mapping** – as explained above.
3. **Intermediate splitting** – the entire process in parallel on different clusters. In order to group them in “Reduce Phase” the similar KEY data should be on the same cluster.
4. **Reduce** – it is nothing but mostly group by phase.
5. **Combining** – The last phase where all the data (individual result set from each cluster) is combined together to form a result.

Conclusion:

Hence we looked at how we can install step of Hadoop and understand simple word application that counts number of access each word in a given input using Hadoop framework an local stead above setup.

Questions:

1. What is the map reduce explain with a small example?
2. Write down steps to install hadoop.

Assignment No.	02 (GROUP B)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 2

Title: Design a distributed application using Map-Reduce which processes a log file of a system.

Prerequisites:

1. Single Node user for First time User
2. Data Analytics with Hadoop

Objectives: Students should be able to perform the distributed application using Map reduce processes a log file of a system .

Theory:

HDFS:

HDFS (Hadoop Distributed File System) is the primary storage system used by Hadoop applications. This open source framework works by rapidly transferring data between nodes. It's often used by companies who need to handle and store big data. HDFS is a key component of many Hadoop systems, as it provides a means for managing big data, as well as supporting big data analytics. There are many companies across the globe that use HDFS, so what exactly is it and why is it needed? Let's take a deep dive into what HDFS is and why it may be useful for businesses.

ARCHITECTURE:

HDFS has been designed to be easily portable from one platform to another. This facilitates widespread adoption of HDFS as a platform of choice for a large set of applications.

NameNode and DataNodes

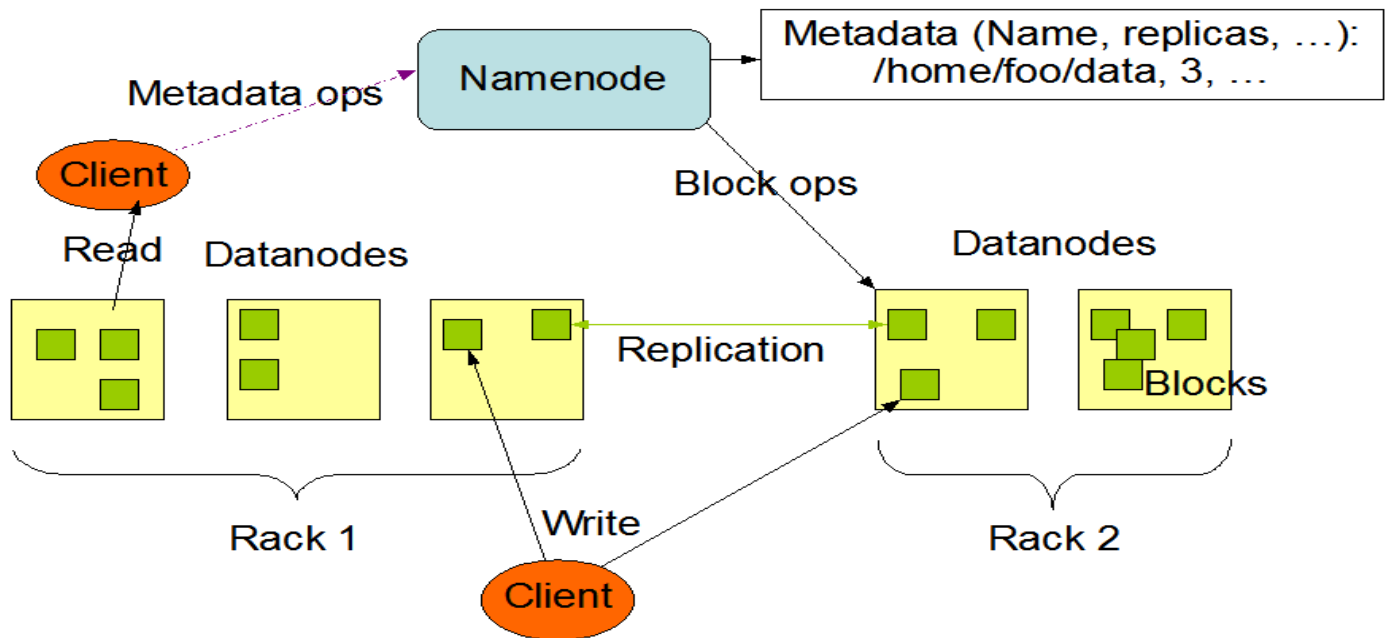
HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

The File System Namespace

HDFS supports a traditional hierarchical file organization. A user or an application can create directories and store files inside these directories. The file system namespace hierarchy is similar to most other existing file systems; one can create and remove files, move a file from one directory to another, or rename a file. HDFS supports user quotas and access permissions. HDFS does not support hard links or soft links. However, the HDFS architecture does not preclude implementing these features.

While HDFS follows naming convention of the FileSystem, some paths and names (e.g. /.reserved and .snapshot) are reserved. Features such as transparent encryption and snapshot use reserved paths.

HDFS Architecture

**Data Replication**

HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file.

All blocks in a file except the last block are the same size, while users can start a new block without filling out the last block to the configured block size after the support for variable length block was added to append and hsync.

Steps to Install Hadoop for distributed environment:

Initially create one folder `logfiles1` on desktop. In that folder store input file (`access_log_short.csv`), `SalesMapper.java`, `SalesCountryReducer.java`, `SalesCountryDriver.java` files)

Step 1) Go to Hadoop home directory and format the NameNode.

```
cd hadoop-2.7.3
```

```
bin/hadoop namenode -format
```

Step 2) Once the NameNode is formatted, go to `hadoop-2.7.3/sbin` directory and start all the daemons/nodes.

```
cd hadoop-2.7.3/sbin
```

1) Start NameNode:

The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.


```
./hadoop-daemon.sh start namenode
```

2) Start DataNode:

On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for different operations.

```
./hadoop-daemon.sh start datanode
```

3) Start Resource Manager:

Resource Manager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each Node Managers and the each application's Application Master.

```
./yarn-daemon.sh start resource manager
```

4) Start Node Manager:

The Node Manager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the Resource Manager.

```
./yarn-daemon.sh start node manager
```

5) Start Job History Server:

JobHistoryServer is responsible for servicing all job history related requests from client.

```
./mr-jobhistory-daemon.sh start historyserver
```

Step 3) To check that all the Hadoop services are up and running, run the below command.

```
jps
```

Step 4) cd

Step 5) sudo mkdir **mapreduce_vijay**

Step 6) sudo chmod 777 -R **mapreduce_vijay/**

Step 7) sudo chown -R **vijay mapreduce_vijay/**

Step 8) sudo cp /home/**vijay**/Desktop/logfiles1/* ~/**mapreduce_vijay/**

Step 9) cd **mapreduce_vijay/**

Step 10) ls

Step 11) sudo chmod +r *.*

Step 12) export CLASSPATH="/home/**vijay**/hadoop-2.7.3/share/hadoop/mapreduce/hadoop-mapreduce-client-core-2.7.3.jar:/home/**vijay**/hadoop-2.7.3/share/hadoop/mapreduce/hadoop-mapreduce-client-common-2.7.3.jar:/home/**vijay**/hadoop-2.7.3/share/hadoop/common/hadoop-common-2.7.3.jar:~/**mapreduce_vijay**/SalesCountry/*:\$HADOOP_HOME/lib/*"

Step 13) javac -d . SalesMapper.java SalesCountryReducer.java SalesCountryDriver.java

Step 14) ls

Step 15) cd SalesCountry/

Step 16) ls (check is class files are created)

Step 17) cd ..

Step 18) gedit Manifest.txt

(add following lines to it:

Main-Class: SalesCountry.SalesCountryDriver)

Step 19) jar -cfm mapreduce_vijay.jar Manifest.txt SalesCountry/*.class

Step 20) ls

Step 21) cd

Step 22) cd mapreduce_vijay/

Step 23) sudo mkdir /input200

Step 24) sudo cp access_log_short.csv /input200

Step 25) \$HADOOP_HOME/bin/hdfs dfs -put /input200 /

Step 26) \$HADOOP_HOME/bin/hadoop jar mapreduce_vijay.jar /input200 /output200

Step 27) hadoop fs -ls /output200

Step 28) hadoop fs -cat /out321/part-00000

Step 29) Now open the Mozilla browser and go to **localhost:50070/dfshealth.html** to check the NameNode interface.

Conclusion:

Hence we have designed distributed application using map reduce which process a log file in a system.

Questions:

1. Write down the steps for Design a distributed application using MapReduce which processes a log file of a system.
2. Write a note on HDFS?
3. Explain Architecture of HDFS.

Assignment No.	03 (GROUP B)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 3

Title:

Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the textinput files and finds average for temperature, dew point and wind speed.

Prerequisites:

1. Concept of Hadoop, Weather Data.
2. Data Analytics with Hadoop

Objectives: Students should be able to perform the weather data is reads the text input file and find the average for temperature, dew point and wind speed using Map Reduce Hadoop .

Theory:**Data analytics:**

Data analytics is the science of analyzing raw data to make conclusions about that information. Many of the techniques and processes of data analytics have been automated into mechanical processes and [algorithms](#) that work over raw data for human consumption.

Understanding Data Analytics

Data analytics is a broad term that encompasses many diverse types of data analysis. Any type of information can be subjected to data analytics techniques to get insight that can be used to improve things. Data analytics techniques can reveal trends and metrics that would otherwise be lost in the mass of information.

Data Analysis Steps: The process involved in data analysis involves several different steps:

1. The first step is to determine the data requirements or how the data is grouped. Data may be separated by age, demographic, income, or gender. Data values may be numerical or be divided by category.
2. The second step in data analytics is the process of collecting it. This can be done through a variety of sources such as computers, online sources, cameras, environmental sources, or through personnel.
3. Once the data is collected, it must be organized so it can be analyzed. This may take place on a spreadsheet or other form of software that can take statistical data.
4. The data is then cleaned up before analysis. This means it is scrubbed and checked to ensure there is no duplication or error, and that it is not incomplete. This step helps correct any errors before it goes on to a data analyst to be analyzed.

What is Hadoop : Hadoop is an open source framework from Apache and is used to store process and analyze data which are very huge in volume. Hadoop is written in Java and is not OLAP (online analytical processing). It is used for batch/offline processing.

Modules of Hadoop:

1. **HDFS:** Hadoop Distributed File System. Google published its paper GFS and on the basis of that HDFS was developed. It states that the files will be broken into blocks and stored in nodes over the distributed architecture.
2. **Yarn:** Yet another Resource Negotiator is used for job scheduling and manage the cluster.

3. **Map Reduce:** This is a framework which helps Java programs to do the parallel computation on data using key value pair. The Map task takes input data and converts it into a data set which can be computed in Key value pair. The output of Map task is consumed by reduce task and then the out of reducer gives the desired result.
4. **Hadoop Common:** These Java libraries are used to start Hadoop and are used by other Hadoop modules.

Weather Data

Weather data provides information about the weather and climate of a region. It tracks patterns and predicts trends. Weather refers to short-term atmospheric conditions of a region and can include indicators such as minimum /maximum temperature, humidity, or wind speed. Climate is the weather of a region averaged over a long period of time. Climate data covers details such as seasonal average temperatures or decade-long patterns of rains and contributes to climate prediction.

Weather data is tremendously important to agriculture and infrastructure planning. Various industries use weather data for real-world business cases, such as travel planning, demand forecast, and supply chain management. The easy availability of weather data for practically any region makes it possible to incorporate it in diverse analytical cases.

What is weather dataset for java, Python?

The Weather Dataset is a **time-series dataset with per-hour information about the weather condition at a particular location**. It records Temperature, dew Point temperature, Relative humidity, Wind speed, Visibility, Pressure and conditions. Workflow: Import library

What types of attributes ?

Common attributes of this type of data include daily weather summary and the following data points:

- Average temperature, median temperature, minimum/maximum temperature, expressed in degree Celsius
- Wind speed in mph or kmph, wind direction
- Humidity in hPa, relative humidity in percentage
- Precipitation including the precipitation types of rain, snow, hail, and sleet, precipitation amount, precipitation probability
- Dew point expressed in degree Celsius or Fahrenheit
- Cloud coverage expressed in oktas
- Ozone information
- UV index
- Pollen concentration per cubic meter
- Daylight hours, Sunrise and sunset time, moon phase
- Solar radiation - typically measured in watts per square meter (w/m2)

different types of weather data?

- **Weather data is a broad data category:** It can be divided into subcategories based on temporal and geographical coverage. Some customers need historical data, while some need real-time data. Some may

focus on a smaller region or a city, while others may want the data for a state or across the globe. For climate change tracking, accurate analysis needs to take into account state, country, region, and global data.

- **Real-time data:** It is typically delivered through real-time weather APIs. This approach makes data available as soon it is updated and provides real-time insights.
- **Historical weather data:** It is available as a downloadable database and also through historical weather APIs. This approach enables access to weather conditions spanning several decades, helping identify patterns and anomalies.
- **Local data:** This data is specific to a small region, a state, or a country. Local weather data in real time is valuable for forecast models which can help the local community for planning their work and making provisions for any natural disasters.
- **Global weather data:** This data provides information on weather and climate trends from across the world. It helps detect climate patterns, track climate change, and power diverse analytical cases.

Conclusion:

Hence we have done dataset for working data which read the text input file and find average for temperature dew point and wind speed.

Questions:

What is map reduce?

Write a short note on map Data Analytics.

Write short note on weather data.

Write down different type weather data.

Assignment No.	04 (GROUP B)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 4

Title: Write a simple program in SCALA using Apache Spark framework.

Prerequisites:

1. Concept of SCALA, Apache Spark Framework.
2. Data Analytics with Hadoop

Objectives: Students should be able to perform a simple program in Scala using Apache Spark framework.

Theory:

SCALA:

Scala stands for Scalable Language. It is a multi-paradigm programming language. Scala language includes features of functional programming and object-oriented programming. It is a statically typed language. Its source code is compiled into bytecode and executed by Java virtual machine(JVM).

Scala uses Java Virtual Machine (JVM) to execute bytecode. Its code is compiled into bytecode and executed by Java Virtual Machine. So you need only JVM to start development with it. Scala can also use all java classes and allows us to create our custom class.

Why use Scala?

It is designed to grow with the demands of its user, from writing small scripts to building a massive system for data processing. Scala is used in Data processing, distributed computing, and web development. It powers the data engineering infrastructure of many companies.

1) Install Scala

Step 1) java -version

Step 2) Install **Scala** from the apt repository by running the following commands to search for scala and install it.

sudo apt search scala ⇒ Search for the package

sudo apt install scala ⇒ Install the package

Step 3) To verify the installation of **Scala**, run the following command.

scala -version

Apache Spark Framework

Apache Spark is a lightning-fast cluster computing technology, designed for fast computation. It is based on Hadoop Map Reduce and it extends the Map Reduce model to efficiently use it for more types of computations, which includes interactive queries and stream processing. The main feature of Spark is its **in-memory cluster computing** that increases the processing speed of an application.

Spark is designed to cover a wide range of workloads such as batch applications, iterative algorithms, interactive queries and streaming. Apart from supporting all these workload in a respective system, it reduces the management burden of maintaining separate tools.

Evolution of Apache Spark

Spark is one of Hadoop's sub project developed in 2009 in UC Berkeley's AMPLab by Matei Zaharia. It was Open Sourced in 2010 under a BSD license. It was donated to Apache software foundation in 2013, and now Apache Spark has become a top level Apache project from Feb-2014.

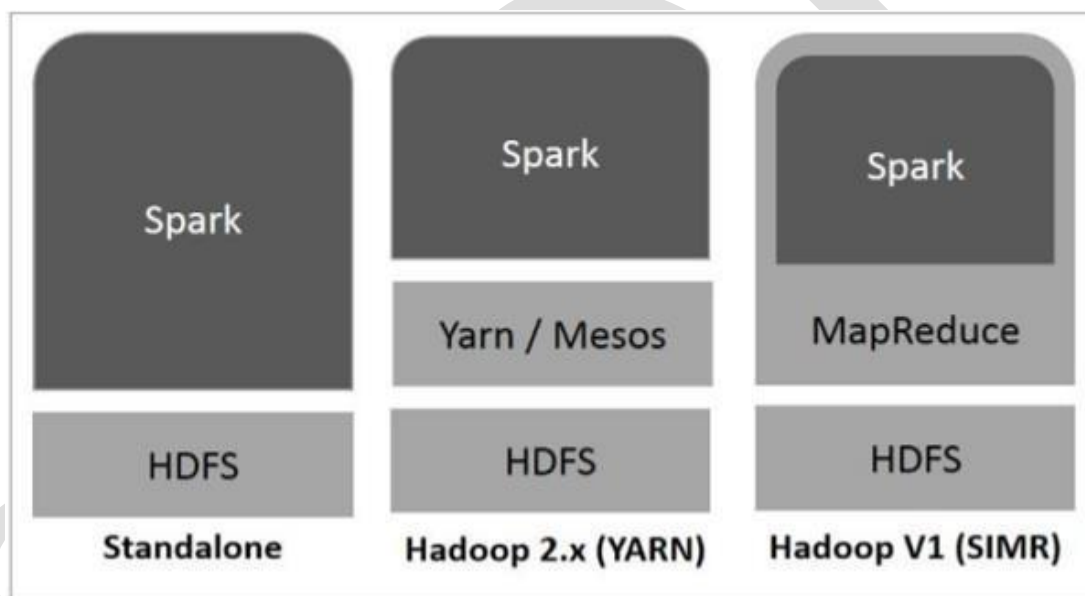
Features of Apache Spark

Apache Spark has following features.

- **Speed** – Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing number of read/write operations to disk. It stores the intermediate processing data in memory.
- **Supports multiple languages** – Spark provides built-in APIs in Java, Scala, or Python. Therefore, you can write applications in different languages. Spark comes up with 80 high-level operators for interactive querying.
- **Advanced Analytics** – Spark not only supports ‘Map’ and ‘reduce’. It also supports SQL queries, Streaming data, Machine learning (ML), and Graph algorithms.

Spark Built on Hadoop

The following diagram shows three ways of how Spark can be built with Hadoop components.

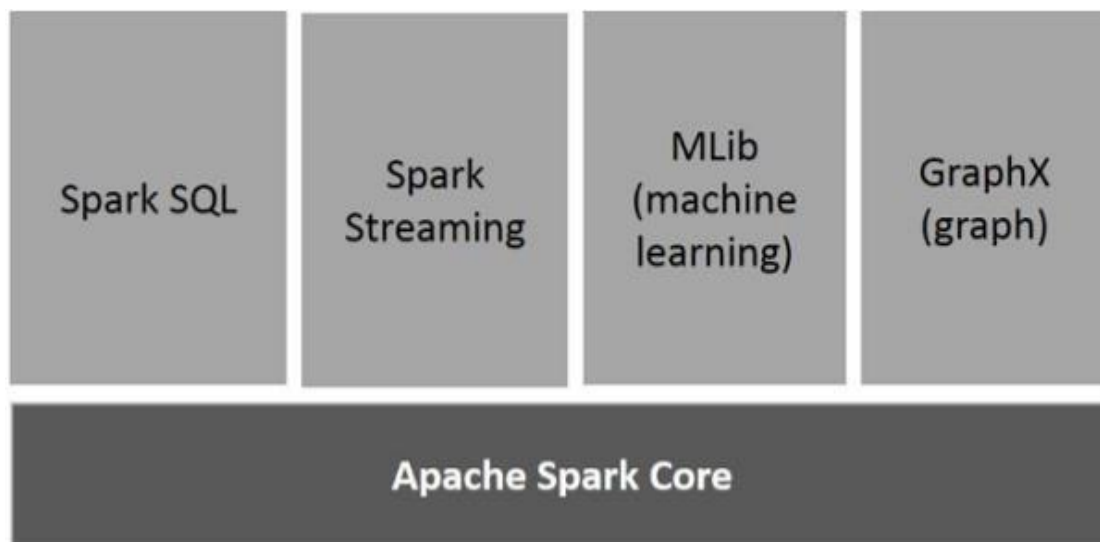


There are three ways of Spark deployment as explained below.

- **Standalone** – Spark Standalone deployment means Spark occupies the place on top of HDFS (Hadoop Distributed File System) and space is allocated for HDFS, explicitly. Here, Spark and MapReduce will run side by side to cover all spark jobs on cluster.
- **Hadoop Yarn** – Hadoop Yarn deployment means, simply, spark runs on Yarn without any pre-installation or root access required. It helps to integrate Spark into Hadoop ecosystem or Hadoop stack. It allows other components to run on top of stack.
- **Spark in MapReduce (SIMR)** – Spark in MapReduce is used to launch spark job in addition to standalone deployment. With SIMR, user can start Spark and uses its shell without any administrative access.

Components of Spark

The following illustration depicts the different components of Spark.



Apache Spark Core

Spark Core is the underlying general execution engine for spark platform that all other functionality is built upon. It provides In-Memory computing and referencing datasets in external storage systems.

Spark SQL

Spark SQL is a component on top of Spark Core that introduces a new data abstraction called SchemaRDD, which provides support for structured and semi-structured data.

Spark Streaming

Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics. It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.

MLlib (Machine Learning Library)

MLlib is a distributed machine learning framework above Spark because of the distributed memory-based Spark architecture. It is, according to benchmarks, done by the MLlib developers against the Alternating Least Squares (ALS) implementations. Spark MLlib is nine times as fast as the Hadoop disk-based version of **Apache Mahout** (before Mahout gained a Spark interface).

GraphX

GraphX is a distributed graph-processing framework on top of Spark. It provides an API for expressing graph computation that can model the user-defined graphs by using Pregel abstraction API. It also provides an optimized runtime for this abstraction.

2) Apache Spark Framework Installation

Apache Spark is an open-source, distributed processing system used for **big data workloads**. It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size.

Step 1) Now go to the official Apache Spark download page and grab the latest version (i.e. 3.2.1) at the time of writing this article. Alternatively, you can use the wget command to download the file directly in the terminal.

wget https://apachemirror.wuchna.com/spark/spark-3.2.1/spark-3.2.1-bin-hadoop2.7.tgz

Step 2) Extract the Apache Spark tar file. `tar -xvzf spark-3.1.1-bin-hadoop2.7.tgz`

Step 3) Move the extracted **Spark** directory to **/opt** directory. `sudo mv spark-3.1.1-bin-hadoop2.7 /opt/spark`

Configure Environmental Variables for Spark

Step 4) Now you have to set a few environmental variables in **.profile** file before starting up the spark.

`echo "export SPARK_HOME=/opt/spark" >> ~/.profile`

`echo "export PATH=$PATH:/opt/spark/bin:/opt/spark/sbin" >> ~/.profile`

`echo "export PYSPARK_PYTHON=/usr/bin/python3" >> ~/.profile`

Step 5) To make sure that these new environment variables are reachable within the shell and available to Apache Spark, it is also mandatory to run the following command to take recent changes into effect.

`source ~/.profile`

Step 6) `ls -l /opt/spark`

Start Apache Spark in Ubuntu

Step 7) Run the following command to start the **Spark** master service and slave service.

`start-master.sh`

`start-workers.sh spark://localhost:7077`

(if workers not starting then remove and install openssh:

`sudo apt-get remove openssh-client openssh-server`

`sudo apt-get install openssh-client openssh-server`)

Step 8) Once the service is started go to the browser and type the following URL access spark page. From the page, you can see my master and slave service is started. <http://localhost:8080/>

Spark Master at spark://LinuxShellTips:7077

URL: spark://LinuxShellTips:7077
 Alive Workers: 1
 Cores in use: 1 Total, 0 Used
 Memory in use: 5.6 GiB Total, 0.0 B Used
 Resources in use:
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-20210501104244-192.168.1.5-39895	192.168.1.5:39895	ALIVE	1 (0 Used)	5.6 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

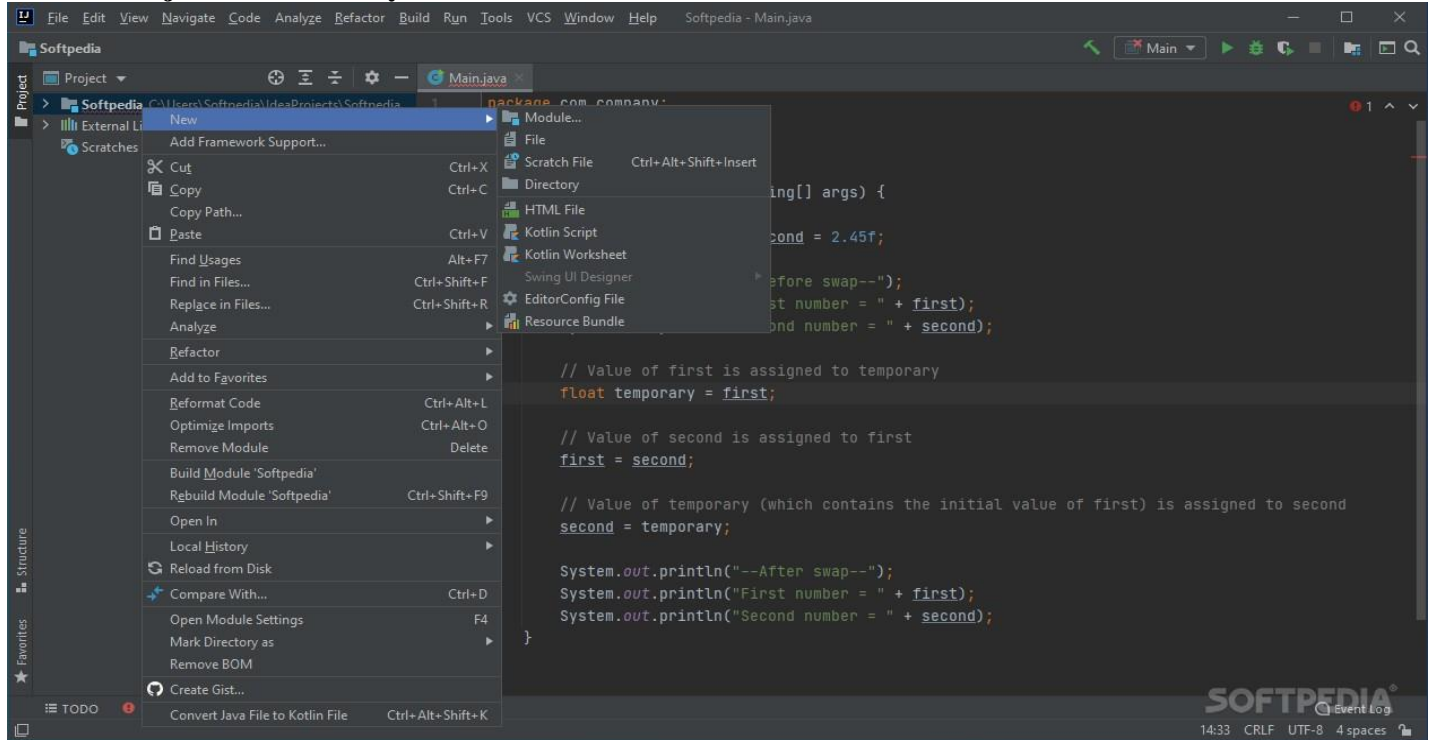
Step 9) You can also check if **spark-shell** works fine by launching the **spark-shell** command.

`Spark-shell`

sudo apt install snapd

snap find "intellij"

Start IntelliJ IDE community Edition



Conclusion:

Hence we have done installation of SCALA and simple program using Apache Spark framework.

Questions:

What is SCALA & Why it is Used?

Write a note on Apache spark Framework.

What is importance of Apache Spark Framework.

GROUP C: MINI PROJECT/CASE STUDY

PYTHON/R

Any 2 mini project are mandatory

Assignment No.	1 (GROUP C) (Mini Project)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No.	2 (GROUP C) (Mini Project)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No.	3 (GROUP C) (Mini Project)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No.	4 (GROUP C) (Mini Project)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No.	5 (GROUP C) (Mini Project)
Title	
Subject	Data Science & Big Data Analytics Laboratory
Class	T.E. (C.E.)
Roll No.	
Date	
Signature	