Sinhgad Institutes

STES's

RMD Sinhgad School of Engineering, Warje-58

310251-Data Science and Big Data Analytics

Department of Computer Engineering

TE (2019 Course)

SINHGAD TECHNICAL EDUCATION SOCIETY'S

RMD SINHGAD SCHOOL OF ENGINEERING

Warje, Pune 411058

Department of Computer Engineering

Sinhgad Institutes

LABORATORY MANUAL

A.Y.:2024-25

DATA SCIENCE AND BIG DATA ANALYTICS LABORATORY

TE Computer Engineering
Semester –II
Subject Code – (310251)

Teaching Scheme	CREDIT	Examination Scheme and Marks
Practical: 04 Hrs / Week	02	Term work: 50 Marks Practical: 25 Marks

Prepared By:

Mrs. Jyoti S. Raghatwan
Mrs. Deepali Bhaturkar
(Assistant Professor, Department of Computer Engineering)

CONTENTS

Group A : Data Science

Data Wrangling I

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

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).
3. Load the Dataset into pandas dataframe.
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.

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.

Data Wrangling II

2. 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.

3. **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.

Provide the codes with outputs and explain everything that you do in this step.

Data Analytics I

4. 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.

The objective is to predict the value of prices of the house using the given features.

5. 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.

6. Data Analytics III

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.

7. Text Analytics

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

8. 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.

9. 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')
2. Write observations on the inference from the above statistics.

10. 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 boxplot for each feature in the dataset.
4. Compare distributions and identify outliers.

Group B- Big Data Analytics – JAVA/SCALA (Any three)

1. Write a code in JAVA for a simple WordCount application that counts the number of occurrences of each word in a given input set using the Hadoop MapReduce framework on local-standalone set-up.

2. Design a distributed application using MapReduce which processes a log file of a system.

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.

4. Write a simple program in SCALA using Apache Spark framework

Group C- Mini Projects/ Case Study – PYTHON/R (Any TWO Mini Project)

1. Write a case study on Global Innovation Network and Analysis (GINA). Components of analytic plan are 1. Discovery business problem framed, 2. Data, 3. Model planning analytic technique and 4. Results and Key findings.

2. Use the following dataset and classify tweets into positive and negative tweets. https://www.kaggle.com/ruchi798/data-science-tweets

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

4. Use the following covid_vaccine_statewise.csv dataset and perform following analytics on the given dataset
https://www.kaggle.com/sudalairajkumar/covid19-in-india?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
e. Number of females vaccinated

5. Write a case study to process data driven for Digital Marketing OR Health care systems with Hadoop Ecosystem components as shown. (Mandatory)
- 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 algorithm libraries
- Solar, Lucene: Searching and Indexing

Group A

Assignment No: 1

Title of the Assignment: Data Wrangling, I

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 insult(), 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.

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

Prerequisite:

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

---

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: 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.

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
```

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.

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

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

1. Dataframe Operations:

Sr. No	Data Frame Function	Description
1	**dataset.head(n=5)**	**Return the first n rows.**
2	**dataset.tail(n=5)**	**Return the last n rows.**
3	**dataset.index**	The index (row labels) of the Dataset.
4	**dataset.columns**	The column labels of the Dataset.
5	**dataset.shape**	Return a tuple representing the dimensionality of the Dataset.
6	**dataset.dtypes**	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	**dataset.columns.values**	Return the columns values in the Dataset in array format
8	**dataset.describe(include='all')**	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	**dataset['Column name]**	Read the Data Column wise.
10	**dataset.sort_index(axis=1, ascending=False)**	Sort object by labels (along an axis).
11	**dataset.sort_values(by="Column name")**	Sort values by column name.
12	**dataset.iloc[5]**	Purely integer-location based indexing for selection by position.
13	**dataset[0:3]**	Selecting via [], which slices the rows.
14	**dataset.loc[:, ["Col_name1", "col_name2"]]**	Selection by label
15	**dataset.iloc[:n, :]**	a subset of the first n rows of the original data

16	**dataset.iloc[:, :n]**	a subset of the first n columns of the original data
17	**dataset.iloc[:m, :n]**	a subset of the first m rows and the first n columns

2. 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	`df.dtypes`
			`sepal length (cm)  float64`
			`sepal width (cm)  float64`
			`petal length (cm)  float64`
			`petal width (cm)  float64`
			`dtype: object`

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)	`df.dtypes`
			`sepal length (cm)  float64`
			`sepal width (cm)  float64`
			`petal length (cm)  int64`
			`petal width (cm)  float64`
			`dtype: object`

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
```

3. **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 t**end to be represented by a non-numeric value.**

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

1. **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.
2. **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.

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

Viva 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?**

Group A

Assignment No: 2

Title of the Assignment: 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.

Objective of the Assignment: Students should be able to perform thedata wrangling operation using Python on any open source dataset

---

Prerequisite:

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

Theory:

1. **Creation of Dataset using Microsoft Excel.**

   The dataset is created in "CSV" format.

   - The name of dataset is **Students Performance**
   - **The features of the dataset are:** Math_Score, Reading_Score, Writing_Score, Placement_Score, Club_Join_Date .
   - **Number of Instance**s: 10

- **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].

2. **Identification and Handling of Null Values**

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.

To facilitate the 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()**

```
df.isnull()
```

   - **Checking for missing values using notnull()**

```
df.notnull()
```

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.

- **Filling null values with a single value**

```
ndf=df
ndf.fillna(0)

data['math score'] = data['math score'].fillna(data['math score'].mean())
```

```
data["math score"] = data["math score"].fillna(data["math score"].median())

data['math score"] = data["math score"].fillna(data["math score"].std())
```

replacing missing values in forenoon column with minimum/maximum number of that column

```
data["math score"] = data["math score"].fillna(data["math score"].min())

data["math score"] = data["math score"].fillna(data["math score"].max())
```

- **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.

  ```
  df.dropna()
  ```

  To Drop rows if all values in that row are missing

  ```
  df.dropna(how = 'all')
  ```

  To Drop columns with at least 1 null value.

  ```
  df.dropna(axis = 1)
  ```

3. Identification and Handling of Outliers

3.1 Identification of Outliers

One of the most important steps as part of data preprocessing 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?

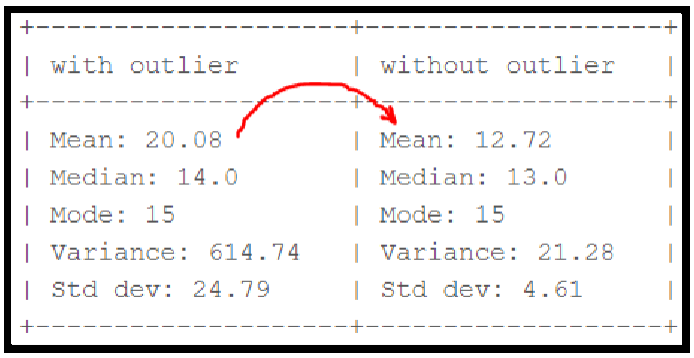
In statistics, we have three measures of central tendency namely Mean, Median, and Mode. They help us describe the data.

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

Fig. Computation with and without outlier

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)

```
col = ['math score', 'reading score' , 'writing
score','placement score']
df.boxplot(col)
```

4.1 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.

4.2 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

4.3 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.

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

Data Transformation: 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 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

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.

Viva 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.**

Group A

Assignment No: 3

Title of the Assignment: 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) variables. 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.

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

Prerequisite:

1. Basic of Python Programming
2. Concept of statistics such as means, median, minimum, maximum, standard deviation etc.

Measures of central tendency

A measure of central tendency (also referred to as measures of centre or central location) is a summary measure that attempts to describe a whole set of data with a single value that represents the middle or center of its distribution.

There are three main measures of central tendency: the mode, the median and the mean. Each of these measures describes a different indication of the typical or central value in the distribution.

1. **Mode**

The mode is the *most commonly occurring value* in a distribution.

Consider this dataset showing the retirement age of 11 people, in whole years:

54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

This table shows a simple frequency distribution of the retirement age data.

Age	Frequency
54	3
55	1
56	1
57	2
58	2
60	2

The most commonly occurring value is 54, therefore the mode of this distribution is 54 years.

Advantage of the mode:

The mode has an advantage over the median and the mean as it can be found for both numerical and categorical (non-numerical) data.

Limitations of the mode:

There are some limitations to using the mode. In some distributions, the mode may not reflect the center of the distribution very well. When the distribution of retirement age is ordered from lowest to highest value, it is easy to see that the center of the distribution is 57 years, but the mode is lower, at 54 years.

54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

It is also possible for there to be more than one mode for the same distribution of data, (bi-modal, or multi-modal). The presence of more than one mode can limit the ability of the mode in describing the center or typical value of the distribution because a single value to describe the center cannot be identified.

In some cases, particularly where the data are continuous, the distribution may have no mode at all (i.e. if all values are different).

In cases such as these, it may be better to consider using the median or mean, or group the data into appropriate intervals, and find the modal class.

2. **Median**

The median is the *middle value* in distribution when the values are arranged in ascending or descending order.

The median divides the distribution in half (there are 50% of observations on either side of the median value). In a distribution with an odd number of observations, the median value is the middle value.

Looking at the retirement age distribution (which has 11 observations), the median is the middle value, which is 57 years:

54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

When the distribution has an even number of observations, the median value is the mean of the two middle values. In the following distribution, the two middle values are 56 and 57, therefore the median equals 56.5 years:

52, 54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

Advantage of the median:

The median is less affected by outliers and skewed data than the mean, and is usually the preferred measure of central tendency when the distribution is not symmetrical.

Limitation of the median:

The median cannot be identified for categorical nominal data, as it cannot be logically ordered.

3. **Mean**

The mean is the sum of the value of each observation in a dataset divided by the number of observations. This is also known as the arithmetic average.

Looking at the retirement age distribution again:

54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

The mean is calculated by adding together all the values (54+54+54+55+56+57+57+58+58+60+60 = 623) and dividing by the number of observations (11) which equals 56.6 years.

Advantage of the mean:

The mean can be used for both continuous and discrete numeric data.

Limitations of the mean:

The mean cannot be calculated for categorical data, as the values cannot be summed.

As the mean includes every value in the distribution the mean is influenced by outliers and skewed distributions.

Variability

The Variance

In a population, **variance** is the average squared deviation from the population mean, as defined by the following formula:

$$\sigma^2 = \Sigma (X_i - \mu)^2 / N$$

Where σ^2 is the population variance, μ is the population mean, X_i is the *i*th element from the population, and N is the number of elements in the population.

Observations from a simple random sample can be used to estimate the variance of a population. For this purpose, sample variance is defined by slightly different formula, and uses a slightly different notation:

$$s^2 = \Sigma (x_i - x)^2 / (n - 1)$$

Where s^2 is the sample variance, x is the sample mean, x_i is the *i*th element from the sample, and n is the number of elements in the sample. Using this formula, the sample variance can be considered an unbiased estimate of the true population variance. Therefore, if you need to estimate an unknown population variance, based on data from a simple random sample, this is the formula to use.

The Standard Deviation

The **standard deviation** is the square root of the variance. Thus, the standard deviation of a population is:

$$\sigma = \text{sqrt}[\sigma^2] = \text{sqrt}[\Sigma (X_i - \mu)^2 / N]$$

Where σ is the population standard deviation, μ is the population mean, X_i is the *i*th element from the population, and N is the number of elements in the population.

Statisticians often use simple random samples to estimate the standard deviation of a population, based on sample data. Given a simple random sample, the best estimate of the standard deviation of a population is:

$$s = \text{sqrt}[s^2] = \text{sqrt}[\Sigma (x_i - x)^2 / (n - 1)]$$

Where *s* is the sample standard deviation, x is the sample mean, x_i is the *i*th element from the sample, and n is the number of elements in the sample.

Conclusion: In this way we have explored the functions of the python library for basic

statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris- versicolor' of iris.csv dataset.

Viva 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.

Group A

Assignment No: 4

Title of the Assignment: 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.
The objective is to predict the value of prices of the house using the given features.

Objective of the Assignment: Students should be able to data analysis using liner regression using Python for any open source dataset

Prerequisite:

1. Basic of Python Programming

2.Concept of Regression.

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 shown as an equation of line like :

        Y = m*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.

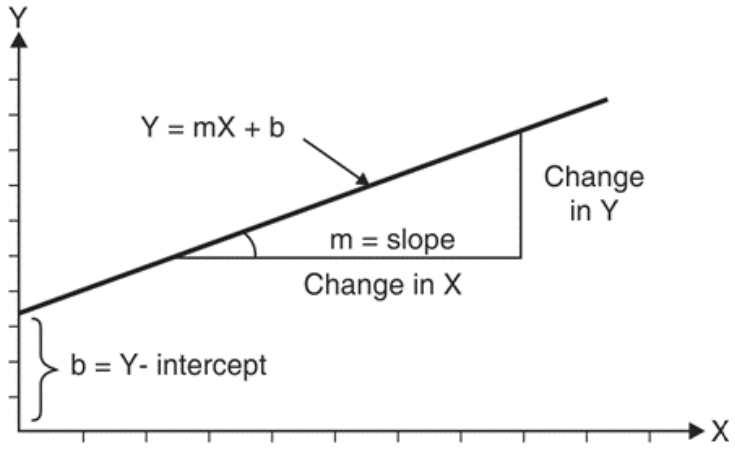
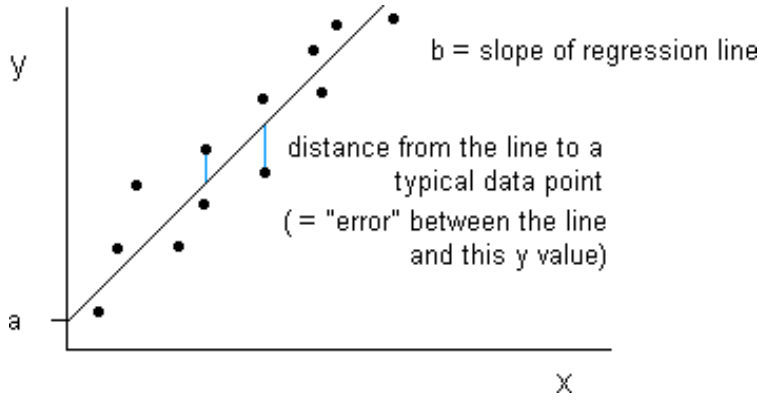
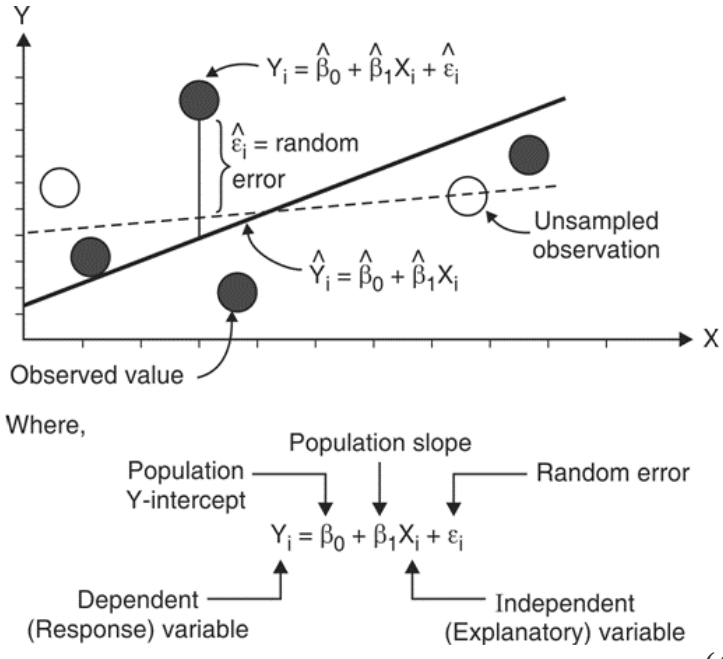


Fig. 1: geometry of linear regression

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



Multivariate 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.



(4)

2. 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.

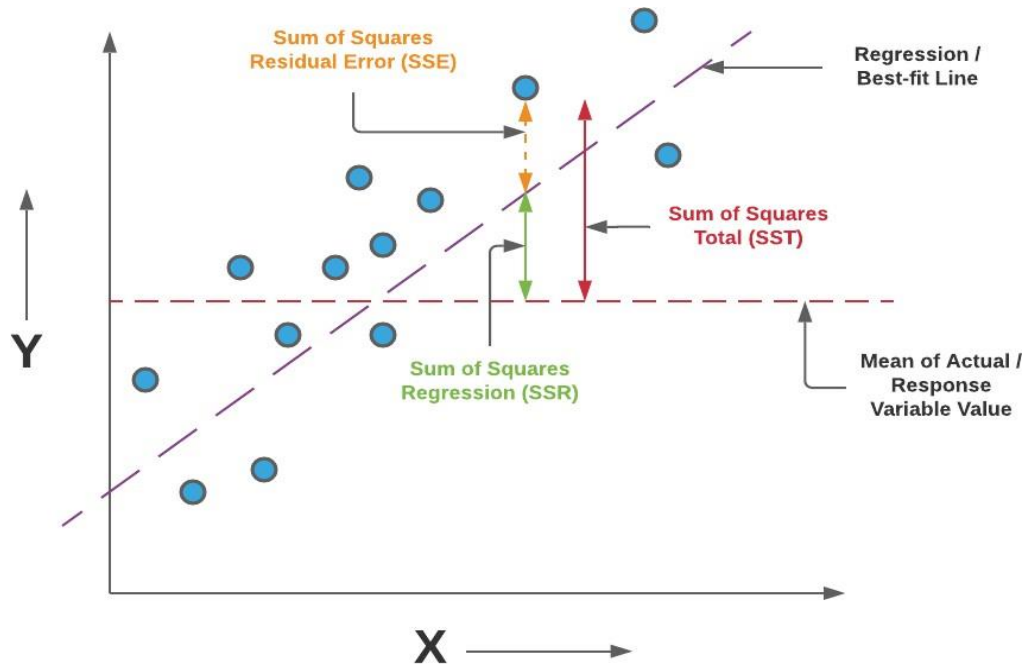
$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

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

RMSE:

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

R-Squared :



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

3. 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.

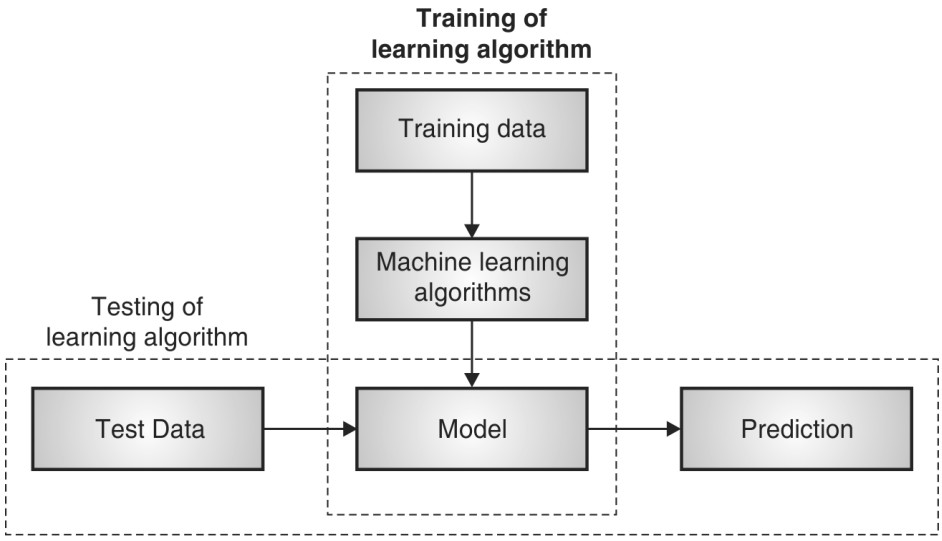


Fig. 1.3.1: Training and Testing Phase in Machine Learning

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.

Viva 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)

.

Group A

Assignment No: 5

Title of the Assignment:

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..

Objective of the Assignment: Students should be able to data analysis using logistic regression using Python for any open source dataset

Prerequisite:

1. Basic of Python Programming

2.Concept of Regression.

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. 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.

   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.

   Linear Regression Equation:

$$y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots\ldots + \beta nXn$$

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

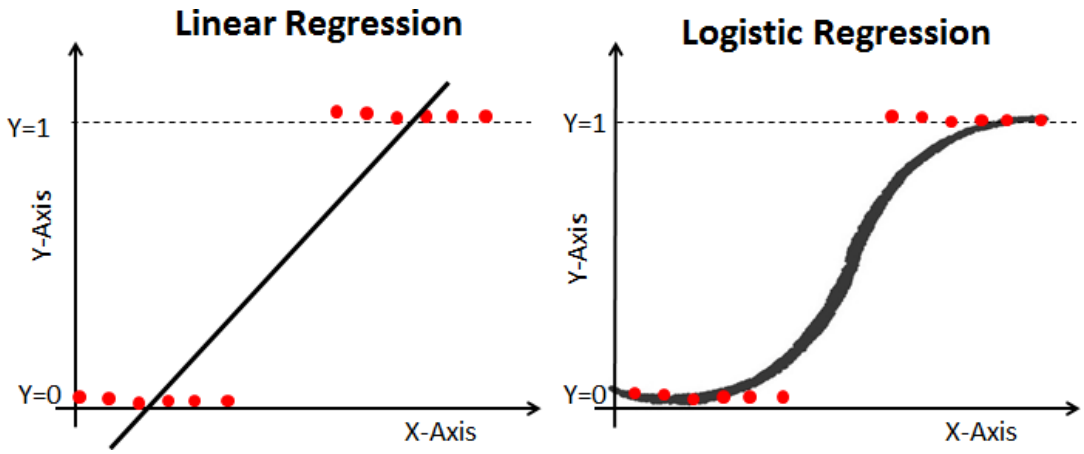
   Sigmoid Function:

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

   Apply Sigmoid function on linear regression:

$$p = 1/1 + e^{-(\beta 0 + \beta 1X1 + \beta 2X2 \ldots\ldots \beta nXn)}$$

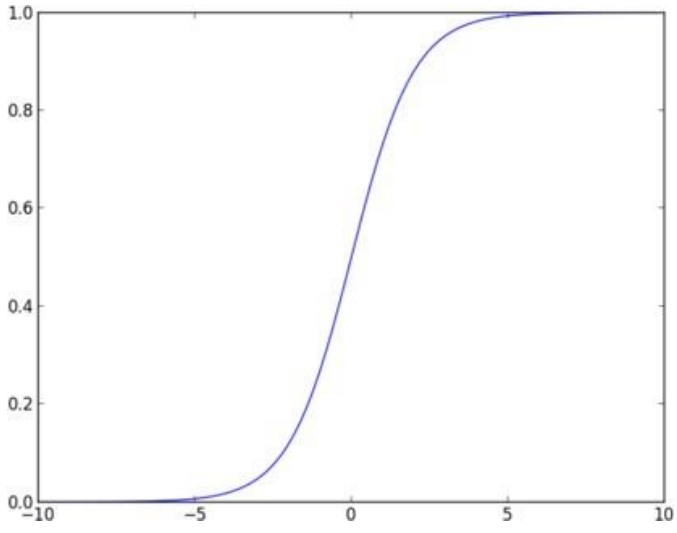
2. Differentiate between Linear and Logistic Regression



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.

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



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.

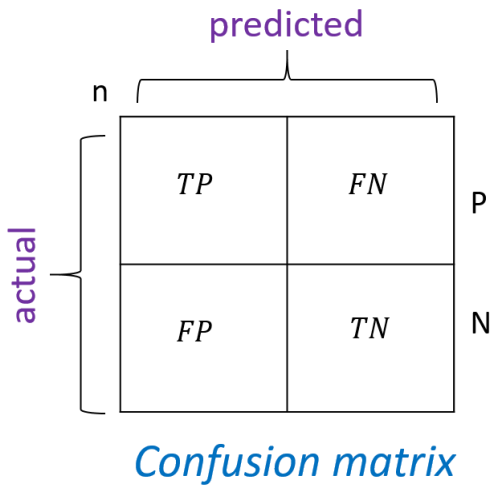
Multinomial Logistic Regression: The target variable has three or more nominal categories such as predicting the type of Wine.

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.



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.

- **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.

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

$$err = 1 - acc$$

- **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 positive instances.

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

Conclusion:

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

Viva 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.**

.

Group A

Assignment No: 6

Title of the Assignment:

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.

Objective of the Assignment: Students should be able to data analysis using Naïve Bayes classification algorithm using Python for any open source dataset

Prerequisite:

1. Basic of Python Programming

2.Concept of Join and Marginal Probability.

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,….j}
    - Let, input observation is specified by 'P' features. Therefore input observation x is given , x = {F1,F2,…..Fp}
    - 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:

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

$$P\left(\frac{B}{A}\right) = \frac{P(B \cap A)}{P(A)} \qquad \ldots\ldots\ldots(2)$$

From equation (1) and (2) ,

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

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

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

$$X = [\underbrace{\text{Outlook}}_{x_1}, \underbrace{\text{Temp}}_{x_2}, \underbrace{\text{Humidity}}_{x_3}, \underbrace{\text{Windy}}_{x_4}]$$

$$C_k = [\underbrace{\text{Yes}}_{C_1}, \underbrace{\text{No}}_{C_2}]$$

Conditional Probability

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

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

Now we have two classes and four features, so if we write this formula for class C1, it will be something like this.

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

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

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

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	?

$$P(Yes \mid X) = P(Rainy \mid Yes) \times P(Cool \mid Yes) \times P(High \mid Yes) \times P(True \mid Yes) \times P(Yes)$$

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

$$P(No \mid X) = P(Rainy \mid No) \times P(Cool \mid No) \times P(High \mid No) \times P(True \mid No) \times P(No)$$

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

P (N0 | Today) > P (Yes | 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

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

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

Step 8: Calculate the required evaluation parameters

Conclusion:

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

Viva 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.

.

Group A

Assignment No: 7

Title of the Assignment:

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

Objective of the Assignment: Students should be able to perform **Text Analysis** using TF IDF Algorithm

Prerequisite:

1. Basic of Python Programming

2. Basic of English language.

1. Basic concepts of Text Analytics

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 metadataNLP helps identify sentiment, finding entities in the sentence, and category of blog/article. Text mining is preprocessed data for text analytics.

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.

- 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. E.g. A stemmer reduces the words like fishing, fished, and fisher to the stem fish.

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

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.

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.

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

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.

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

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 completed text data analysis using TF IDF algorithm

Viva Questions:

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.**

2) **Write Python code for removing stop words from the below documents, conver 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'
```

.

Group A

Assignment No: 8

Title of the Assignment: 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.

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

Prerequisite:

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

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 ahigh-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.

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

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

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 customise the scatter plot into a hexagonal plot, where, the more the colour intensity, the more will be the number of observations.

c. The Rug Plot

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.

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.

a. 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.

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.

b. 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:

c. 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.

If there are any outliers or the passengers that do not belong to any of the quartiles, they are called outliers and are represented by dots on the box plot.

d. 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 violinplot() 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.

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.

The stripplot() 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.

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.

1. 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.

To plot matrix plots, we need useful information on both columns and row headers. One way to do this is to call the corr() method on the dataset. The corr() function returns the correlation between all the numeric columns of the dataset.

```
dataset.corr()
```

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.

4. **Checking how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram.**

```
import seaborn as sns

dataset = sns.load_dataset('titanic')

sns.histplot(dataset['fare'], kde=False, bins=10)
```

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.

Assignment 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?**

Group A
Assignment No: 9

Title of the Assignment: 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')
2. Write observations on the inference from the above statistics.

Objective of the Assignment: Students should be able to perform the data Visualizationoperation using Python on any open source dataset

Prerequisite:

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

Theory:

BoxPlot:

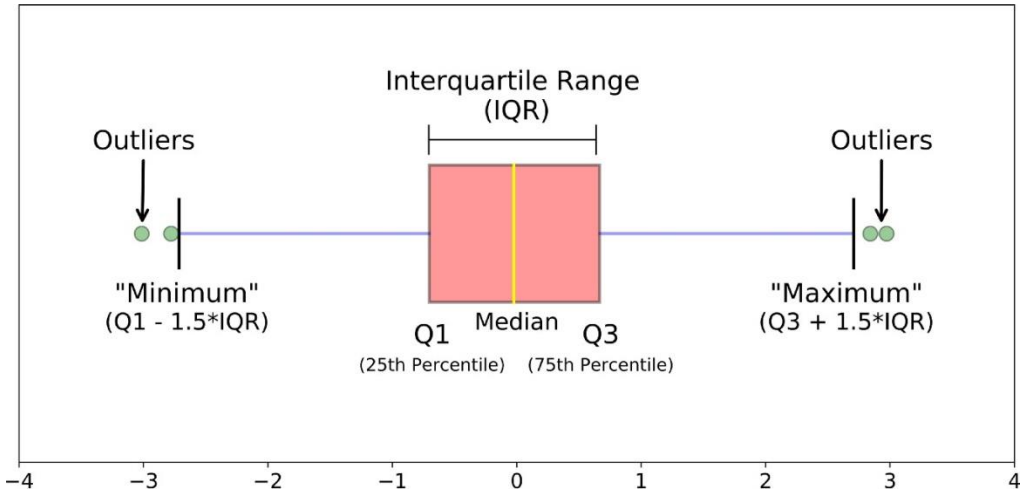
A boxplot is a standardized way of displaying the distribution of data based on a five number summary ("minimum", first quartile (Q1), median, third quartile (Q3), and "maximum"). It can tell you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed.

median (Q2/50th Percentile): the middle value of the dataset.

first quartile (Q1/25th Percentile): the middle number between the smallest number (not the "minimum")and the median of the dataset.

Third quartile (Q3/75th Percentile): the middle value between the median and the highest value (not the"maximum") of the dataset.

Interquartile range (IQR): 25th to the 75th percentile.



Algorithm:

1. **Import the required libraries**

   import seaborn as sns

2. **Create the datafame of inbuilt dataset titanic**

   dataset =

   sns.load_dataset('titanic')

   dataset

3. **Plot a box plot for distribution of age with respect to each gender along with the informationabout whether theysurvived or not.**

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

   sns.boxplot(x='sex', y='age', data =

   dataset,hue='survived')

4. **Write the observation from the displayed box plot.**

Viva Questions

1. Explain following plots with sample diagrams:
   - Histogram
   - Violin Plot
2. Write code to plot a box plot for distribution of age with respect to each genderalong with the information about whether they survived or not. Write the observations.

Group A

Assignment No: 10

Title of the Assignment: 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.

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

Prerequisite:

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

Theory:

Histograms:

A histogram is basically used to represent data provided in a form of some groups.It is accurate method for the graphical representation of numerical data distribution. It is a type of bar plot where X-axis represents the bin ranges while Y-axis gives information about frequency.

The following table shows the parameters accepted by matplotlib.pyplot.hist() function :

Attribute	Parameter
x	array or sequence of array
bins	optional parameter contains integer or sequence or strings
density	optional parameter contains boolean values
range	optional parameter represents upper and lower range of bins
histtype	optional parameter used to create type of histogram [bar, barstacked, step, stepfilled], default is“bar”
align	optional parameter controls the plotting of histogram [left, right, mid]
weights	optional parameter contains array of weights having same dimensions as x
bottom	location of the basline of each bin
rwidth	optional parameter which is relative width of the bars with respect to bin width
color	optional parameter used to set color or sequence of color specs
label	optional parameter string or sequence of string to match with multiple datasets
log	optional parameter used to set histogram axis on log scale

Algorithm:

1. **Import required libraries.**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import pylab

import seaborn as sns

import os

2. **Create the data frame for downloaded iris.csv dataset.**

os.chdir("D:\Pandas")

df =

pd.read_csv("Iris.csv")df

3. **Apply data preprocessing techniques.**

   df.isnull().sum()

   df.describe()

4. **Plot the box plot for each feature in the dataset and observe and detect the outliers.***sns.set(style ="whitegrid", palette = "GnBu_d", rc = {'figure.figsize':(11.7,8.27)}) sns.boxplot(x='Species', y='SepalLengthCm', data=df) plt.title('Distribution of sepal length')*

   plt.show()

5. **Plot the histogram for each feature in the dataset.**

   df.hist()

Viva 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

Assignment No: 1

Title of the Assignment: Big Data Analytics I

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

Objective of the Assignment: Students should be able to perform the Word Count application using the Hadoop Map-Reduce framework.

Theory:

Steps to Install Hadoop

Steps to install Hadoop:

Step 1) mkdir words

Step 2) Download hadoop-core-1.2.1.jar, which is used to compile and execute the program. Visit the following link
http://mvnrepository.com/artifact/org.apache.hadoop/hadoop-core/1.2.1

Step 3) Put that downloaded jar file into words folder

Step 4) Implement WordCount.java program

Step 5) Create inputl.txt on home directory with some random text

Step 6) go on words path then compile

javac -classpath/home/vijay/words/hadoop-core-1.2.1.jar/home/vijay/words/WordCount.java

Step 7) jar -cvf words.jar -c words/
Step 8) cd.. then use following commands

hadoop fs -mkdir/input

hadoop fs -put input1.txt/input

hadoop fs –ls/input

hadoop jar/home/vijay/words/words12.jar WordCount/input/input1.txt/out321

hadoop fs –ls/out321

hadoop fs –cat/out321/part-r-00000

(Otherwise check in Browsing HDFS→ Utilities → Browse the file System→/)

Conclusion: In this way we have explored the program in JAVA for a simple Word Count application that counts the number occurrences of each word in a given input set using the Hadoop Map-Reduce framework on local-standalone set-up.

Viva Questions

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

Group B

Assignment No: 2

Title of the Assignment: Big Data Analytics I

Design a distributed application using MapReduce which processes a log file of a system.

Theory:

- **Steps to Install Hadoop for distributed environment**

Steps to Install Hadoop for distributed environment:

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

Step 1) Go to the 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 files 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 ResourceManager:

ResourceManager 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 NodeManagers and each application's ApplicationMaster.

./yarn-daemon.sh start resourcemanager

4) Start NodeManager:

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

./yarn-daemon.sh start nodemanager

5) Start JobHistoryServer:

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: In this way we have designed a distributed application using MapReduce which processes a log file of a system

Viva Questions

1. Write down the steps for Design a distributed application using MapReduce which processes a log file of a system.

Group B

Assignment No: 3

Title of the Assignment: Big Data Analytics I

Write a simple program in SCALA using Apache Spark framework.

Theory:

- **Steps to Install Scala**
- **Apache Spark Framework Installation**

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

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

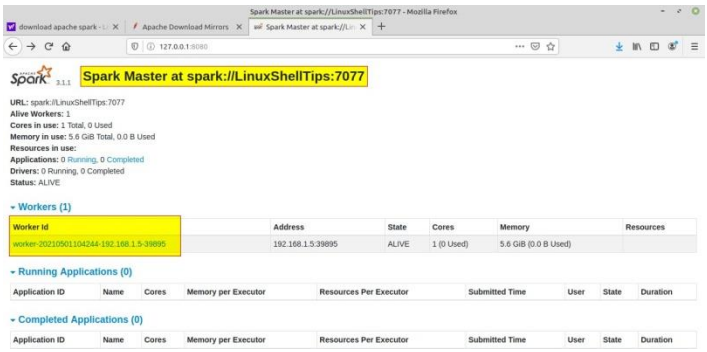
Step 7) Run the following command to start the **Spark** master service and slave service.

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/



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”

sudo snap install intellij-idea-community - - classic

Start Intellij IDE community Edition

Conclusion: In this way we have explored the program in SCALA using Apache Spark framework

Viva Questions

1. **Write down steps to install Scala**.

Group C

Assignment No: 2

Contents for Theory:

1. **Step 1: Data Collection**
2. **Step 2: Sentiment Analysis**
3. **Step 3: Visualization.**

We will begin by scraping and storing Twitter data. We will then classify the Tweets into positive, negative, or neutral sentiment with a simple algorithm. Then, we will build charts using Plotly and Matplotlib to identify trends in sentiment.

Step 1: Data collection

Command -

```
import pandas as pd
df = pd.read_csv('/content/data_visualization.csv')
```

Output -

```
/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshe
ll.py:2882: DtypeWarning: Columns (22,24) have mixed types.Specify
dtype option on import or set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

Let's now take a look at some of the variables present in the data frame:

Command -

```
df.info()
```

Output -

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33590 entries, 0 to 33589
Data columns (total 36 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               33590 non-null  int64
 1   conversation_id  33590 non-null  int64
 2   created_at       33590 non-null  object
 3   date             33590 non-null  object
```

```
 4   time              33590 non-null  object
 5   timezone          33590 non-null  int64
 6   user_id           33590 non-null  int64
 7   username          33590 non-null  object
 8   name              33590 non-null  object
 9   place             85 non-null     object
 10  tweet             33590 non-null  object
 11  language          33590 non-null  object
 12  mentions          33590 non-null  object
 13  urls              33590 non-null  object
 14  photos            33590 non-null  object
 15  replies_count     33590 non-null  int64
 16  retweets_count    33590 non-null  int64
 17  likes_count       33590 non-null  int64
 18  hashtags          33590 non-null  object
 19  cashtags          33590 non-null  object
 20  link              33590 non-null  object
 21  retweet           33590 non-null  bool
 22  quote_url         1241 non-null   object
 23  video             33590 non-null  int64
 24  thumbnail         9473 non-null   object
 25  near              0 non-null      float64
 26  geo               0 non-null      float64
 27  source            0 non-null      float64
 28  user_rt_id        0 non-null      float64
 29  user_rt           0 non-null      float64
 30  retweet_id        0 non-null      float64
 31  reply_to          33590 non-null  object
 32  retweet_date      0 non-null      float64
 33  translate         0 non-null      float64
 34  trans_src         0 non-null      float64
 35  trans_dest        0 non-null      float64
dtypes: bool(1), float64(10), int64(8), object(17)
memory usage: 9.0+ MB
```

The data frame has 35 columns. The most main variables we will be using in this analysis are date and tweet. Let's take a look at a sample Tweet in this dataset, and see if we can predict whether it is positive or negative:

Command -

```
df['tweet'][10]
```

Output -

```
We are pleased to invite you to the EDHEC DataViz Challenge grand
final for a virtual exchange with all Top 10 finalists to see how
data visualization creates impact and can bring out compelling
stories in support of @UNICEF’s mission. https://t.co/Vbj9B48VjV
```

Step 2: Sentiment Analysis

The Tweet above is clearly positive. Let's see if the model is able to pick up on this, and return a positive prediction. Run the following lines of code to import the NLTK library, along with the SentimentIntensityAnalyzer (SID) module.

Command -

```
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
import re
import pandas as pd
import nltk
nltk.download('words')
words = set(nltk.corpus.words.words())
```

The SID module takes in a string and returns a score in each of these four categories - positive, negative, neutral, and compound. The compound score is calculated by normalizing the positive, negative, and neutral scores. If the compound score is closer to 1, then the Tweet can be classified as positive. If it is closer to -1, then the Tweet can be classified as negative. Let's now analyze the above sentence with the sentiment intensity analyzer.

Command -

```
sentence = df['tweet'][0]
sid.polarity_scores(sentence)['compound']
```

The output of the code above is 0.7089, indicating that the sentence is of positive sentiment. Let's now create a function that predicts the sentiment of every Tweet in the dataframe, and stores it as a separate column called 'sentiment.' First, run the following lines of code to clean the Tweets in the data frame:

Command –

```
def cleaner(tweet):
    tweet = re.sub("@[A-Za-z0-9]+","",tweet) #Remove @ sign
    tweet = re.sub(r"(?:\@|http?\://|https?\://|www)\S+", "",
tweet) #Remove http links
    tweet = " ".join(tweet.split())
    tweet = tweet.replace("#", "").replace("_", " ") #Remove
hashtag sign but keep the text
    tweet = " ".join(w for w in nltk.wordpunct_tokenize(tweet)
         if w.lower() in words or not w.isalpha())
```

```
    return tweet
df['tweet_clean'] = df['tweet'].apply(cleaner)
```

Now that the Tweets are cleaned, run the following lines of code to perform the sentiment analysis:

Command -

```
word_dict =
{'manipulate':-1,'manipulative':-1,'jamescharlesiscancelled':-1,'j
amescharlesisoverparty':-1,

'pedophile':-1,'pedo':-1,'cancel':-1,'cancelled':-1,'cancel
culture':0.4,'teamtati':-1,'teamjames':1,
            'teamjamescharles':1,'liar':-1}
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
sid.lexicon.update(word_dict)
list1 = []
for i in df['tweet_clean']:
    list1.append((sid.polarity_scores(str(i)))['compound'])
```

The word_dict created above is a dictionary of custom words I wanted to add into the model. Words like 'teamjames' mean that people's sentiment around James Charles is positive, and that they support him. The dictionary used to train the sentiment intensity analyzer wouldn't already have these words in them, so we can update it ourselves with custom words.

Now, we need to convert the compound scores into categories - 'positive', 'negative', and 'neutral.'

Command -

```
df['sentiment'] = pd.Series(list1)
def sentiment_category(sentiment):
    label = ''
    if(sentiment>0):
        label = 'positive'
    elif(sentiment == 0):
        label = 'neutral'
    else:
        label = 'negative'
    return(label)
df['sentiment_category'] =
df['sentiment'].apply(sentiment_category)
```

Let's take a look at the head of the data frame to ensure everything is working properly:

Command -

```
df = df[['tweet','date','id','sentiment','sentiment_category']]
df.head()
```

Output -

	tweet	date	id	sentiment	sentiment_category
0	Take your storytelling to the next level using...	2021-06-20	1406335989484822531	0.7089	positive
1	Choosing Fonts for Your Data Visualization | b...	2021-06-19	1406292636789526537	0.0000	neutral
2	This data visualization shows where our greate...	2021-06-19	1406082288035811330	0.0000	neutral
3	Looking for examples of stellar charts made so...	2021-06-18	1405948260796100610	0.4019	positive
4	With #WISQARS Data Visualization, you can disp...	2021-06-18	1405942146960613376	-0.4215	negative

Notice that the first few Tweets are the combination of positive, negative and neutral sentiment. For this analysis, we will only be using Tweets with positive and negative sentiment, since we want to visualize how stronger sentiments have changed over time.

Step 3: Visualization

Now that we have Tweets classified as positive and negative, let's take a look at changes in sentiment over time. We first need to group positive and negative sentiment and count them by date:

Command -

```
neg = df[df['sentiment_category']=='negative']
neg = neg.groupby(['date'],as_index=False).count()
pos = df[df['sentiment_category']=='positive']
pos = pos.groupby(['date'],as_index=False).count()
pos = pos[['date','id']]
neg = neg[['date','id']]
```

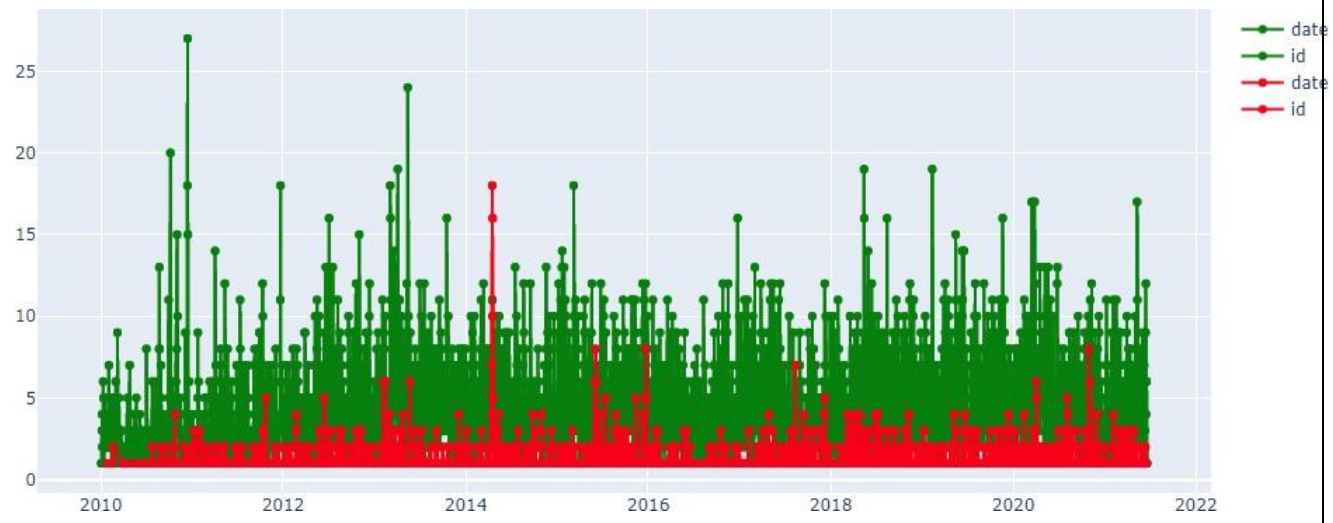
Now, we can visualize sentiment by date using Plotly, by running the following lines of code:

Command -

```
import plotly.graph_objs as go
fig = go.Figure()
for col in pos.columns:
    fig.add_trace(go.Scatter(x=pos['date'],  y=pos['id'],
                             name = col,
                             mode = 'markers+lines',
                             line=dict(shape='linear'),
                             connectgaps=True,
                             line_color='green'
```

```
                                )
                    )
for col in neg.columns:
    fig.add_trace(go.Scatter(x=neg['date'],  y=neg['id'],
                                name = col,
                                mode = 'markers+lines',
                                line=dict(shape='linear'),
                                connectgaps=True,
                                line_color='red'
                                )
                    )
fig.show()
```

Final Output - You should see a chart that looks like this:



The red line represents negative sentiment, and the green line represents positive sentiment.

Assignment Questions:

1. **What is Twitter sentiment analysis?**
2. **What is Natural Language Processing (NLP) and what are the stages in the life cycle of NLP?**
3. **What is NLTK? How to tokenize a sentence using the NLTK package?**
4. **Explain any two real-life applications of Natural Language Processing.**

Group C

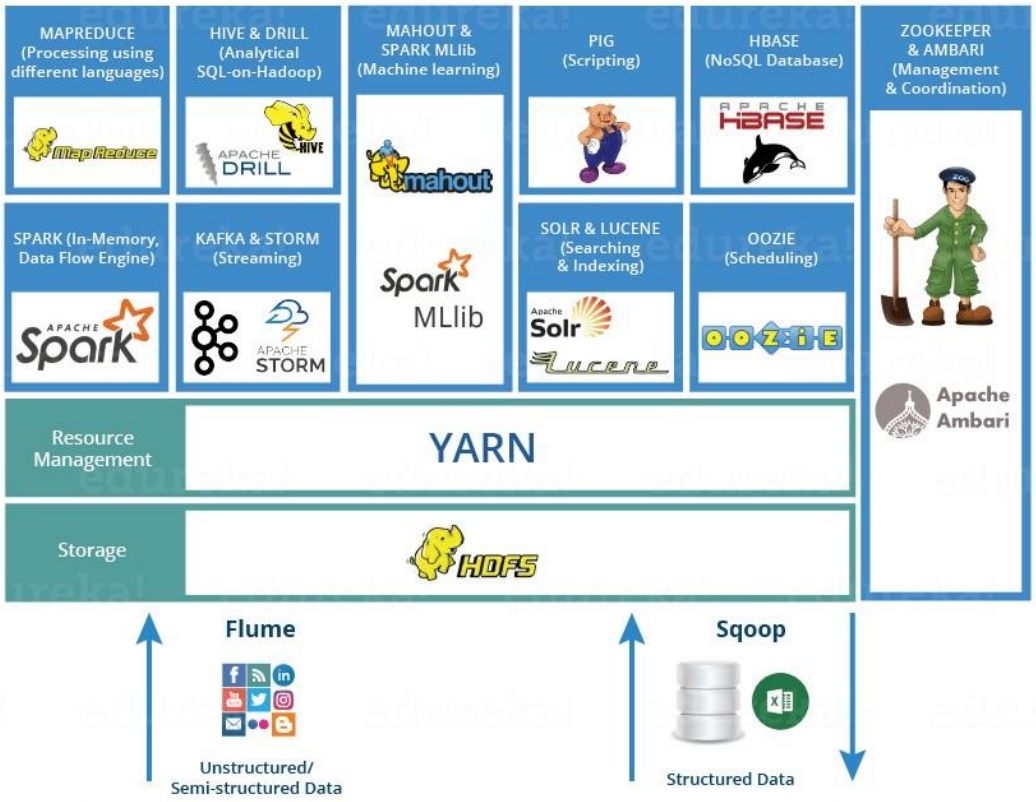
Assignment No: 5

Contents for Theory:

1. **Hadoop Ecosystem**
2. **HDFS**
3. **YARN**
4. **MAPREDUCE**
5. **APACHE PIG**
6. **APACHE HIVE**
7. **APACHE MAHOUT**
8. **APACHE SPARK**
9. **APACHE HBASE**
10. **APACHE SOLR & LUCENE**

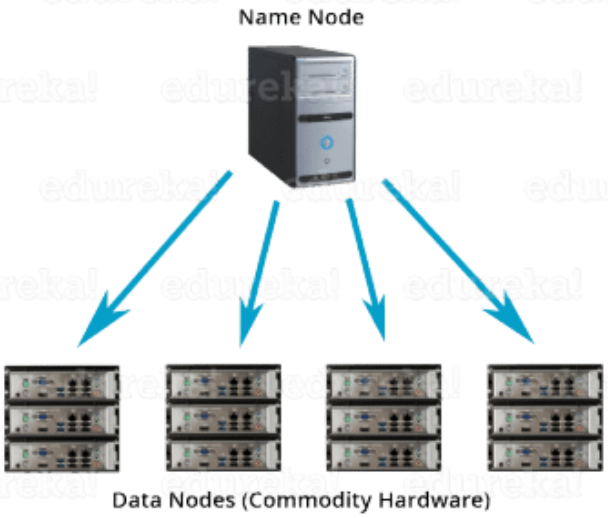
Hadoop Ecosystem:

Hadoop Ecosystem is neither a programming language nor a service, it is a platform or framework which solves big data problems. You can consider it as a suite which encompasses a number of services (ingesting, storing, analyzing and maintaining) inside it. Let us discuss and get a brief idea about how the services work individually and in collaboration. Below are the Hadoop components, that together form a Hadoop ecosystem,



HDFS -

- *Hadoop Distributed File System* is the core component or you can say, the backbone of Hadoop Ecosystem.
- HDFS is the one, which makes it possible to store different types of large data sets (i.e. structured, unstructured and semi structured data).
- HDFS creates a level of abstraction over the resources, from where we can see the whole HDFS as a single unit.
- It helps us in storing our data across various nodes and maintaining the log file about the stored data (metadata).
- HDFS has two core components, i.e. NameNode and DataNode.
    1. The NameNode is the main node and it doesn't store the actual data. It contains metadata, just like a log file or you can say as a table of content. Therefore, it requires less storage and high computational resources.
    2. On the other hand, all your data is stored on the DataNodes and hence it requires more storage resources. These DataNodes are commodity hardware (like your laptops and desktops) in the distributed environment. That's the reason, why Hadoop solutions are very cost effective.
    3. You always communicate to the NameNode while writing the data. Then, it internally sends a request to the client to store and replicate data on various DataNodes.



YARN - Consider YARN as the brain of your Hadoop Ecosystem. It performs all your processing activities by allocating resources and scheduling tasks.

- It has two major components, i.e. ResourceManager and NodeManager.

1. ResourceManager is again a main node in the processing department.
2. It receives the processing requests, and then passes the parts of requests to corresponding NodeManagers accordingly, where the actual processing takes place.
3. NodeManagers are installed on every DataNode. It is responsible for execution of task on every single DataNode.

1. Schedulers: Based on your application resource requirements, Schedulers perform scheduling algorithms and allocates the resources.
2. ApplicationsManager: While ApplicationsManager accepts the job submission, negotiates to containers (i.e. the Data node environment where process executes) for executing the application specific ApplicationMaster and monitoring the progress.

   ApplicationMasters are the deamons which reside on DataNode and communicates to containers for execution of tasks on each DataNode.ResourceManager has two components, i.e. Schedulers and ApplicationsManager.

MAPREDUCE -

It is the core component of processing in a Hadoop Ecosystem as it provides the logic of processing. In other words, MapReduce is a software framework which helps in writing applications that processes large data sets using distributed and parallel algorithms inside Hadoop environment.

- In a MapReduce program, Map() and Reduce() are two functions.
  1. The Map function performs actions like filtering, grouping and sorting.
  2. While Reduce function aggregates and summarizes the result produced by map function.

3. The result generated by the Map function is a key value pair (K, V) which acts as the input for Reduce function.

Student	Department	Count	(Key, Value), Pair
Student 1	D1	1	(D1, 1)
Student 2	D1	1	(D1, 1)
Student 3	D1	1	(D1, 1)
Student 4	D2	1	(D2, 1)
Student 5	D2	1	(D2, 1)
Student 6	D3	1	(D3, 1)
Student 7	D3	1	(D3, 1)

Let us take the above example to have a better understanding of a MapReduce program. We have a sample case of students and their respective departments. We want to calculate the number of students in each department. Initially, Map program will execute and calculate the students appearing in each department, producing the key value pair as mentioned above. This key value pair is the input to the Reduce function. The Reduce function will then aggregate each department and calculate the total number of students in each department and produce the given result.

Department	Total Student
D1	3
D2	2
D3	2

APACHE PIG -

- PIG has two parts: Pig Latin, the language and the pig runtime, for the execution environment. You can better understand it as Java and JVM.
- It supports *pig latin* language, which has SQL like command structure.

As everyone does not belong from a programming background. So, Apache PIG relieves them. *You might be curious to know how?*

Well, an interesting fact is:
10 line of pig latin = approx. 200 lines of Map-Reduce Java code

But don't be shocked when I say that at the back end of Pig job, a map-reduce job executes.

- The compiler internally converts pig latin to MapReduce. It produces a sequential set of MapReduce jobs, and that's an abstraction (which works like black box).
- PIG was initially developed by Yahoo.
- It gives you a platform for building data flow for ETL (Extract, Transform and Load), processing and analyzing huge data sets.

How Pig works?
In PIG, first the load command, loads the data. Then we perform various functions on it like grouping, filtering, joining, sorting, etc. At last, either you can dump the data on the screen or you can store the result back in HDFS.

APACHE HIVE -

- Facebook created HIVE for people who are fluent with SQL. Thus, HIVE makes them feel at home while working in a Hadoop Ecosystem.
- Basically, HIVE is a data warehousing component which performs reading, writing and managing large data sets in a distributed environment using SQL-like interface.

HIVE + SQL = HQL

- The query language of Hive is called Hive Query Language(HQL), which is very similar like SQL.
- It has 2 basic components: Hive Command Line and JDBC/ODBC driver.
- The Hive Command line interface is used to execute HQL commands.
- While, Java Database Connectivity (JDBC) and Object Database Connectivity (ODBC) is used to establish connection from data storage.
- Secondly, Hive is highly scalable. As, it can serve both the purposes, i.e. large data set processing (i.e. Batch query processing) and real time processing (i.e. Interactive query processing).
- It supports all primitive data types of SQL.
- You can use predefined functions, or write tailored user defined functions (UDF) also to accomplish your specific needs.

APACHE MAHOUT -

Now, let us talk about Mahout which is renowned for machine learning. Mahout provides an environment for creating machine learning applications which are scalable.

So, What is machine learning?

Machine learning algorithms allow us to build self-learning machines that evolve by itself without being explicitly programmed. Based on user behavior, data patterns and past experiences it makes important future decisions. You can call it a descendant of Artificial Intelligence (AI).

What Mahout does?

It performs collaborative filtering, clustering and classification. Some people also consider frequent item sets missing as Mahout's function. Let us understand them individually:

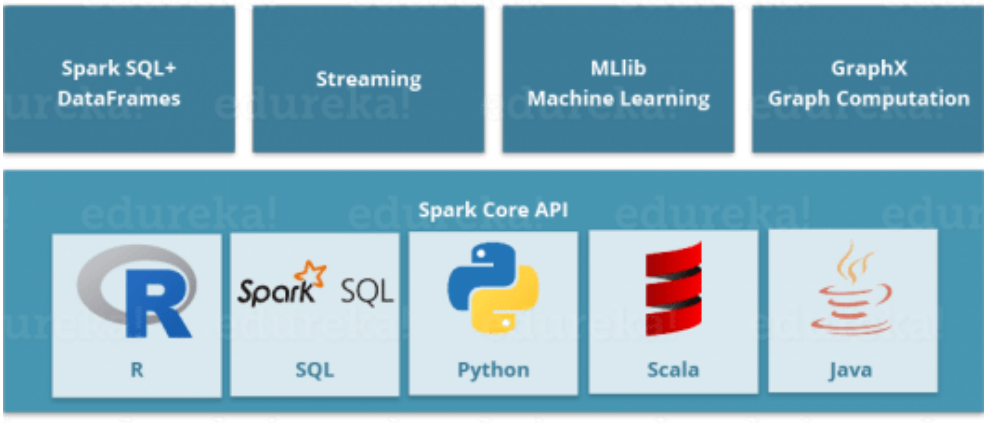
1. Collaborative filtering: Mahout mines user behaviors, their patterns and their characteristics and based on that it predicts and make recommendations to the users. The typical use case is an E-commerce website.

2. Clustering: It organizes a similar group of data together like articles can contain blogs, news, research papers etc.
3. Classification: It means classifying and categorizing data into various sub-departments like articles can be categorized into blogs, news, essay, research papers and other categories.
4. Frequent item set missing: Here Mahout checks, which objects are likely to be appearing together and make suggestions, if they are missing. For example, cell phone and cover are brought together in general. So, if you search for a cell phone, it will also recommend you the cover and cases.

Mahout provides a command line to invoke various algorithms. It has a predefined set of library which already contains different inbuilt algorithms for different use cases.

APACHE SPARK -

- Apache Spark is a framework for real time data analytics in a distributed computing environment.
- The Spark is written in Scala and was originally developed at the University of California, Berkeley.
- It executes in-memory computations to increase speed of data processing over Map-Reduce.
- It is 100x faster than Hadoop for large scale data processing by exploiting in-memory computations and other optimizations. Therefore, it requires high processing power than Map-Reduce.



As you can see, Spark comes packed with high-level libraries, including support for R, SQL, Python, Scala, Java etc. These standard libraries increase the seamless integrations in complex workflow. Over this, it also allows various sets of services to integrate with it like MLlib, GraphX, SQL + Data Frames, Streaming services etc. to increase its capabilities.

This is a very common question in everyone's mind:

"Apache Spark: A Killer or Saviour of Apache Hadoop?" – O'Reily

The Answer to this – This is not an apple to apple comparison. Apache Spark best fits for real time processing, whereas Hadoop was designed to store unstructured data and execute batch processing over it. When we combine, Apache Spark's ability, i.e. high processing speed, advance analytics and multiple integration support with Hadoop's low cost operation on commodity hardware, it gives the best results.

That is the reason why, Spark and Hadoop are used together by many companies for processing and analyzing their Big Data stored in HDFS.

APACHE HBASE -

- HBase is an open source, non-relational distributed database. In other words, it is a NoSQL database.
- It supports all types of data and that is why, it's capable of handling anything and everything inside a Hadoop ecosystem.
- It is modelled after Google's BigTable, which is a distributed storage system designed to cope up with large data sets.
- The HBase was designed to run on top of HDFS and provides BigTable like capabilities.
- It gives us a fault tolerant way of storing sparse data, which is common in most Big Data use cases.
- The HBase is written in Java, whereas HBase applications can be written in REST, Avro and Thrift APIs.

APACHE SOLR & LUCENE -

Apache Solr and Apache Lucene are the two services which are used for searching and indexing in the Hadoop Ecosystem.

- Apache Lucene is based on Java, which also helps in spell checking.
- If Apache Lucene is the engine, Apache Solr is the car built around it. Solr is a complete application built around Lucene.
- It uses the Lucene Java search library as a core for search and full indexing.

Assignment Questions -

1. Why is there a need for Big Data Analytics in Healthcare?
2. What is the role of Hadoop in Healthcare Analytics?
3. Explain how an IBM Watson platform can be used for healthcare analytics?