WT and DA

@Slip – 1

Q. 1) Write a PHP script to keep track of number of times the web page has been accessed (Use Session Tracking).

Ans:

```
<?php
Session_start();

If(isset($_SESSION['pcount'])) {
  $_SESSION['pcount] += 1;
} else {
  $_SESSION['pcount] = 1;
}

Echo "You have visited this page ".$_SESSION['pcount]." Time(s).";
?>
```

Q. 2)Create 'Position_Salaries' Data set. Build a linear regression model by identifying independent and Target variable. Split the variables into training and testing sets. Then divide the training and testing sets Into a 7:3 ratio, respectively and print them. Build a simple linear regression model.

Ans:

```
Import numpy as np
Import pandas as pd
From sklearn.model_selection import train_test_split
From sklearn.linear_model import LinearRegression

# Create the Position_Salaries dataset
Data = {'Position': ['CEO', 'charman', 'director', 'Senior Manager', 'Junior Manager', 'Intern'],
    'Level': [1, 2, 3, 4, 5, 6],
    'Salary': [50000, 80000, 110000, 150000, 200000, 250000]}
Df = pd.DataFrame(data)

# Identify the independent and target variables
X = df.iloc[:, 1:2].values
Y = df.iloc[:, 2].values

# Split the variables into training and testing sets with a 7:3 ratio
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

# Print the training and testing sets
Print("X_train:\n", X_train)
Print("y_train:\n", y_train)
Print("X_test:\n", X_test)
Print("y_test:\n", y_test)

# Build a simple linear regression model
Regressor = LinearRegression()
Regressor.fit(X_train, y_train)

# Print the coefficients and intercept
Print("Coefficients:", regressor.coef_)
Print("Intercept:", regressor.intercept_)
```

@Slip-2

Q. 1Write a PHP script to change the preferences of your web page like font style, font size, font color, Background color using cookie. Display selected setting on next web page and actual implementation (with new settings) on third page (Use Cookies).

Ans :

Fristpage.html

```
<!DOCTYPE html>
<html>
<head>
	<title>Change preferences</title>
</head>
<body>
	<h1>Change preferences</h1>
	<form action=”secondpage.php” method=”post”>
		<label for=”fontstyle”>Font Style:</label>
		<select name=”fontstyle” id=”fontstyle”>
			<option value=”Arial”>Arial</option>
			<option value=”Times New Roman”>Times New Roman</option>
			<option value=”Verdana”>Verdana</option>
		</select><br><br>
		<label for=”fontsize”>Font Size:</label>
		<select name=”fontsize” id=”fontsize”>
			<option value=”12”>12</option>
			<option value=”14”>14</option>
```

```
                        <option value=”16”>16</option>
                </select><br><br>
                <label for=”fontcolor”>Font Color:</label>
                <input type=”color” name=”fontcolor” id=”fontcolor”><br><br>
                <label for=”bgcolor”>Background Color:</label>
                <input type=”color” name=”bgcolor” id=”bgcolor”><br><br>
                <input type=”submit” name=”submit” value=”Save”>
        </form>
</body>
</html>
```

Secondpage.php

```
<?php
If(isset($_POST[‘submit’])) {
        $fontstyle = $_POST[‘fontstyle’];
        $fontsize = $_POST[‘fontsize’];
        $fontcolor = $_POST[‘fontcolor’];
        $bgcolor = $_POST[‘bgcolor’];

        // Set the cookie values
        Setcookie(‘fontstyle’, $fontstyle, time()+86400);
        Setcookie(‘fontsize’, $fontsize, time()+86400);
        Setcookie(‘fontcolor’, $fontcolor, time()+86400);
        Setcookie(‘bgcolor’, $bgcolor, time()+86400);

        // Redirect to the next page
        Header(‘Location: thirdpage.php’);
        Exit();
}
?>
```

Thirdpage.php

```
<?php
// Retrieve the cookie values
$fontstyle = isset($_COOKIE['fontstyle']) ? $_COOKIE['fontstyle'] : 'Arial';
$fontsize = isset($_COOKIE['fontsize']) ? $_COOKIE['fontsize'] : '12';
$fontcolor = isset($_COOKIE['fontcolor']) ? $_COOKIE['fontcolor'] : '#000000';
$bgcolor = isset($_COOKIE['bgcolor']) ? $_COOKIE['bgcolor'] : '#FFFFFF';
?>

<!DOCTYPE html>
<html>
<head>
        <title>Page with new settings</title>
        <style type="text/css">
                Body {
                        Font-family: <?php echo $fontstyle ?>;
                        Font-size: <?php echo $fontsize ?>px;
                        Color: <?php echo $fontcolor ?>;
                        Background-color: <?php echo $bgcolor ?>;
                }
        </style>
</head>
<body>
        <h1>Page with new settings</h1>
        <p>This is the page with the new settings. The font style is <?php echo $fontstyle ?>, the font size is <?php echo $fontsize ?>px, the font color is <?php echo $fontcolor ?>, and the background color is <?php echo $bgcolor ?>.</p>
</body>
</html>
```

Q. 2)Create 'Salary' Data set . Build a linear regression model by identifying independent and target Ariable. Split the variables into training and testing sets and print them. Build a simple linear regression Del for predicting purchases.

Ans:

```
Import numpy as np
Import pandas as pd
From sklearn.model_selection import train_test_split
From sklearn.linear_model import LinearRegression

# Create the Salary dataset
Data = {'YearsExperience': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Salary': [50000, 60000, 70000, 80000, 90000, 100000, 110000, 120000, 130000, 140000]}
Df = pd.DataFrame(data)

# Identify the independent and target variables
X = df.iloc[:, 0:1].values
Y = df.iloc[:, 1].values

# Split the variables into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

```
# Print the training and testing sets
Print("X_train:\n", X_train)
Print("y_train:\n", y_train)
Print("X_test:\n", X_test)
Print("y_test:\n", y_test)

# Build a simple linear regression model
Regressor = LinearRegression()
Regressor.fit(X_train, y_train)

# Print the coefficients and intercept
Print("Coefficients:", regressor.coef_)
Print("Intercept:", regressor.intercept_)
```

@Slip-3

Q. 1) Write a PHP script to accept username and password. If in the first three chances, username and Password entered is correct then display second form with "Welcome message" otherwise display error Message. [Use Session]

.

Ans:

```
<?php
// Start session
Session_start();
```

```
// Check if login form has been submitted
If(isset($_POST['submit'])) {
 // Get username and password input from user
 $username = $_POST['username'];
 $password = $_POST['password'];

 // Set correct username and password
 $correct_username = 'myusername';
 $correct_password = 'mypassword';

 // Check if entered username and password are correct
 If($username == $correct_username && $password == $correct_password) {
  // Set session variable to mark user as logged in
  $_SESSION['loggedin'] = true;

  // Redirect user to welcome page
  Header('Location: welcome.php');
  Exit;
 } else {
  // Decrement login attempts
  If(isset($_SESSION['attempts'])) {
   $_SESSION['attempts']--;
  } else {
   $_SESSION['attempts'] = 3;
  }

  // Display error message if maximum login attempts exceeded
  If($_SESSION['attempts'] <= 0) {
   Echo "Maximum login attempts exceeded. Please try again later.";
```

```
    } else {
      // Display error message
      Echo “Invalid username or password. You have “.$_SESSION[‘attempts’].” Attempt(s) left.”;
    }
  }
}
?>

<!—HTML form for user input →
<form method=”post”>
  <label for=”username”>Username:</label>
  <input type=”text” id=”username” name=”username” required><br><br>

  <label for=”password”>Password:</label>
  <input type=”password” id=”password” name=”password” required><br><br>

  <input type=”submit” name=”submit” value=”Log In”>
</form>
```

Q. 2)Create ‘User’ Data set having 5 columns namely: User ID, Gender, Age, Estimated Salary and urchased. Build a logistic regression model that can predict whether on the given parameter a person will buy a car or not.

Ans:

```
Import pandas as pd

Data = {‘User ID’: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
        ‘Gender’: [‘Male’, ‘Male’, ‘Female’, ‘Female’, ‘Male’, ‘Male’, ‘Female’, ‘Female’, ‘Male’, ‘Female’],
        'Age': [19, 35, 26, 27, 19, 27, 32, 25, 33, 45],
        'Estimated Salary': [19000, 20000, 43000, 57000, 76000, 58000, 82000, 32000, 69000, 65000],
        'Purchased': [0, 0, 0, 1, 1, 0, 1, 0, 1, 1]}
Df = pd.DataFrame(data)

From sklearn.model_selection import train_test_split

X = df.iloc[:, 1:4].values
Y = df.iloc[:, 4].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
From sklearn.linear_model import LogisticRegression

Lr=LogisticRegression(random_state=0)
Lr.fit(X_train, y_train)

# Predict a single observation
Observation = [[0, 30, 87000]]
Prediction = Lr.predict(observation)
Print(prediction)

# Predict multiple observations
Observations = [[0, 30, 87000], [1, 50, 45000], [1, 22, 30000]]
Predictions = Lr.predict(observations)
Print(predictions)
```

@Slip-4

Q. 1) Write a PHP script to accept Employee details (Eno, Ename, Address) on first page. On second Page accept earning (Basic, DA, HRA). On third page print Employee information (Eno, Ename, Address, Basic, DA, HRA, Total) [Use Session]

.

Ans:

Firstpage.php

```
<?php
Session_start();
?>

<!DOCTYPE html>
<html>
<head>
	<title>Employee Details</title>
</head>
<body>
	<h1>Employee Details</h1>
	<form method="POST" action="Secondpage.php">
		<label for="eno">Employee No:</label>
		<input type="text" id="eno" name="eno"><br><br>
		<label for="ename">Employee Name:</label>
		<input type="text" id="ename" name="ename"><br><br>
		<label for="address">Address:</label>
		<textarea id="address" name="address"></textarea><br><br>
		<input type="submit" value="Next">
	</form>
```

```
</body>
</html>

<?php
// Store employee details in session
If(isset($_POST['eno']) && isset($_POST['ename']) && isset($_POST['address'])) {
	$_SESSION['eno'] = $_POST['eno'];
	$_SESSION['ename'] = $_POST['ename'];
	$_SESSION['address'] = $_POST['address'];
}
?>
```

Secondpage.php

```
<?php
Session_start();
?>

<!DOCTYPE html>
<html>
<head>
	<title>Earnings</title>
</head>
<body>
	<h1>Earnings</h1>
	<form method="POST" action="thirdpage.php">
		<label for="basic">Basic:</label>
		<input type="text" id="basic" name="basic"><br><br>
		<label for="da">DA:</label>
```

```
                <input type=”text” id=”da” name=”da”><br><br>
                <label for=”hra”>HRA:</label>
                <input type=”text” id=”hra” name=”hra”><br><br>
                <input type=”submit” value=”Next”>
        </form>
</body>
</html>

<?php
// Store earnings in session
If(isset($_POST[‘basic’]) && isset($_POST[‘da’]) && isset($_POST[‘hra’])) {
        $_SESSION[‘basic’] = $_POST[‘basic’];
        $_SESSION[‘da’] = $_POST[‘da’];
        $_SESSION[‘hra’] = $_POST[‘hra’];
}
?>
```

Thirdpage.php

```
<?php
Session_start();

// Calculate total earnings
$total = $_SESSION[‘basic’] + $_SESSION[‘da’] + $_SESSION[‘hra’];
?>

<!DOCTYPE html>
<html>
```

```
<head>
        <title>Employee Information</title>
</head>
<body>
        <h1>Employee Information</h1>
        <p><strong>Employee No:</strong> <?php echo $_SESSION['eno']; ?></p>
        <p><strong>Employee Name:</strong> <?php echo $_SESSION['ename']; ?></p>
        <p><strong>Address:</strong> <?php echo $_SESSION['address']; ?></p>
        <p><strong>Basic:</strong> <?php echo $_SESSION['basic']; ?></p>
        <p><strong>DA:</strong> <?php echo $_SESSION['da']; ?></p>
        <p><strong>HRA:</strong> <?php echo $_SESSION['hra']; ?></p>
        <p><strong>Total Earnings:</strong> <?php echo $total; ?></p>
</body>
</html>
```

Q. 2)Build a simple linear regression model for Fish Species Weight Prediction.

Ans:

```
Import pandas as pd
Import random
From sklearn.linear_model import LinearRegression

# create the dataset
Fish_species = ['Tuna', 'Salmon', 'Trout', 'Bass', 'Sardine', 'Cod', 'Mackerel']
Weights = []

For i in range(50):
   Fish_weight = []
```

```
  For j in range(7):
    Weight = random.randint(1, 20)
    Fish_weight.append(weight)
  Weights.append(fish_weight)

Df = pd.DataFrame(weights, columns=fish_species)

# create the linear regression model
X = df.iloc[:, :-1] # independent variables
Y = df.iloc[:, -1] # target variable

Model = LinearRegression()
Model.fit(X, y)

# predict the weight of a new fish species
New_fish = [[10, 12, 15, 7, 4, 8]] # example input
Predicted_weight = model.predict(new_fish)
Print("Predicted weight:", predicted_weight)
```

@Slip-5

Q. 1) Create XML file named "Item.xml"with item-name, item-rate, item quantity Store the details of 5 Items of different Types.

Ans:

Item.xml

```
<items>
 <item type="Electronics">
```

```
    <name>Television</name>
    <rate>500</rate>
    <quantity>10</quantity>
  </item>
  <item type=”Clothing”>
    <name>Shirt</name>
    <rate>50</rate>
    <quantity>20</quantity>
  </item>
  <item type=”Grocery”>
    <name>Rice</name>
    <rate>40</rate>
    <quantity>30</quantity>
  </item>
  <item type=”Books”>
    <name>Harry Potter and the Philosopher’s Stone</name>
    <rate>20</rate>
    <quantity>50</quantity>
  </item>
  <item type=”Sports”>
    <name>Football</name>
    <rate>100</rate>
    <quantity>5</quantity>
  </item>
</items>
```

Q. 2)Use the iris dataset. Write a Python program to view some basic statistical details like percentile, Mean, std etc. Of the species of ‘Iris-setosa’, ‘Iris-versicolor’ and ‘Iris-virginica’. Apply logistic regression

On the dataset to identify different species (setosa, versicolor, verginica) of Iris flowers given just 4 Features: sepal and petal lengths and widths.. Find the accuracy of the model.

Ans:

```
Import pandas as pd

From sklearn.datasets import load_iris

From sklearn.linear_model import LogisticRegression

From sklearn.model_selection import train_test_split

From sklearn.metrics import accuracy_score


# load the iris dataset

Iris = load_iris()


# create a dataframe from the dataset

Df = pd.DataFrame(iris.data, columns=iris.feature_names)

Df['target'] = iris.target


# view basic statistical details of the different species

Print("Statistical details of Iris-setosa:")

Print(df[df['target']==0].describe())


Print("Statistical details of Iris-versicolor:")

Print(df[df['target']==1].describe())


Print("Statistical details of Iris-virginica:")

Print(df[df['target']==2].describe())


# split the data into training and testing sets

X = df.iloc[:,:-1]
```

```
Y = df.iloc[:, -1]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# fit a logistic regression model
Logreg = LogisticRegression()
Logreg.fit(X_train, y_train)

# make predictions on the test set
Y_pred = logreg.predict(X_test)

# calculate the accuracy of the model
Accuracy = accuracy_score(y_test, y_pred)
Print("Accuracy of the logistic regression model:", accuracy)
```

@Slip-6

Q. 1) Write PHP script to read "book.xml" file into simpleXML object. Display attributes and elements .

(simple_xml_load_file() function)

.

Ans:

```
<?php
// Load the XML file into a SimpleXML object
$xml = simplexml_load_file("book.xml");

// Display the attributes and elements of the SimpleXML object
Echo "Book attributes: <br>";
Echo "ISBN: " . $xml['isbn'] . "<br>";
Echo “Publisher: “ . $xml[‘publisher’] . “<br>”;
Echo “<br>”;

Echo “Book elements: <br>”;
Echo “Title: “ . $xml->title . “<br>”;
Echo “Author: “ . $xml->author . “<br>”;
Echo “Description: “ . $xml->description . “<br>”;
?>
```

Book.xml file

```
<?xml version=”1.0”?>
<book isbn=”978-3-16-148410-0” publisher=”Example Publisher”>
  <title>Example Book</title>
  <author>John Doe</author>
  <description>This is an example book.</description>
</book>
```

Q. 2)Create the following dataset in python & Convert the categorical values into numeric format.Apply

The apriori algorithm on the above dataset to generate the frequent itemsets and association rules. Repeat

Te process with different min_sup values.

TID={1:[“bread”,”milk”],2=[“bread”,”diaper”,”beer”,”eggs”],3=[“milk”,”diaper”,”beer”,”coke”],4=[“bread”,”milk”,”diaper”,”beer”],5=[“bread”,”milk”,”diaper”,”coke”]}

Ans:

```
Import pandas as pd
```

```
From mlxtend.preprocessing import TransactionEncoder

From mlxtend.frequent_patterns import apriori, association_rules


# create the dataset

TID =
{1:["bread","milk"],2:["bread","diaper","beer","eggs"],3:["milk","diaper","beer","coke"],4:["bread","mil
k","diaper","beer"],5:["bread","milk","diaper","coke"]}

Transactions = []

For key, value in TID.items():

   Transactions.append(value)


# convert the categorical values into numeric format

Te = TransactionEncoder()

Te_ary = te.fit_transform(transactions)

Df = pd.DataFrame(te_ary, columns=te.columns_)


# apply the apriori algorithm with different min_sup values

Min_sup_values = [0.2, 0.4, 0.6]

For min_sup in min_sup_values:

   Frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)

   Rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)

   Print("Min_sup:", min_sup)

   Print("Frequent Itemsets:")

   Print(frequent_itemsets)

   Print("Association Rules:")

   Print(rules)
```

@Slip-7

Q. 1) Write a PHP script to read “Movie.xml” file and print all MovieTitle and ActorName of file using OMDocument Parser. “Movie.xml” file should contain following information with at least 5 records Wth values. M vieInfoMovieNo, MovieTitle, ActorName ,ReReleaseYear.

Ans:

Php file

```
<?php
// Load the XML file
$xml = new DOMDocument();
$xml->load(‘Movie.xml’);

// Get all the movie nodes
$movies = $xml->getElementsByTagName(‘MovieInfo’);

// Loop through each movie node and print the movie title and actor name

Foreach ($movies as $movie) {
   Echo “Movie Title: “ . $movie->getElementsByTagName(‘MovieTitle’)[0]->textContent . “, “;
   Echo “Actor Name: “ . $movie->getElementsByTagName(‘ActorName’)[0]->textContent . “<br>”;
}
?>
```

XML file

```
<?xml version=”1.0”?>
```

```
<MovieList>
  <MovieInfo>
    <MovieNo>1</MovieNo>
    <MovieTitle>The Shawshank Redemption</MovieTitle>
    <ActorName>Tim Robbins</ActorName>
    <ReleaseYear>1994</ReleaseYear>
  </MovieInfo>
  <MovieInfo>
    <MovieNo>2</MovieNo>
    <MovieTitle>The Godfather</MovieTitle>
    <ActorName>Marlon Brando</ActorName>
    <ReleaseYear>1972</ReleaseYear>
  </MovieInfo>
  <MovieInfo>
    <MovieNo>3</MovieNo>
    <MovieTitle>The Dark Knight</MovieTitle>
    <ActorName>Christian Bale</ActorName>
    <ReleaseYear>2008</ReleaseYear>
  </MovieInfo>
  <MovieInfo>
    <MovieNo>4</MovieNo>
    <MovieTitle>The Godfather: Part II</MovieTitle>
    <ActorName>Al Pacino</ActorName>
    <ReleaseYear>1974</ReleaseYear>
  </MovieInfo>
  <MovieInfo>
    <MovieNo>5</MovieNo>
    <MovieTitle>12 Angry Men</MovieTitle>
    <ActorName>Henry Fonda</ActorName>
```

```
        <ReleaseYear>1957</ReleaseYear>
    </MovieInfo>
</MovieList>
```

Q. 2)Download the Market basket dataset. Write a python program to read the dataset and display its Information. Preprocess the data (drop null values etc.) Convert the categorical values into numeric Format. Apply the apriori algorithm on the above dataset to generate the frequent itemsets and association Rules. .

Ans:

```
Import pandas as pd
From mlxtend.preprocessing import TransactionEncoder
From mlxtend.frequent_patterns import apriori, association_rules

# read the dataset
Df = pd.read_csv('Market_Basket_Optimisation.csv', header=None)

# drop null values
Df.dropna(inplace=True)

# convert categorical values to numeric using one-hot encoding
Te = TransactionEncoder()
Te_ary = te.fit(df.values).transform(df.values)
Df = pd.DataFrame(te_ary, columns=te.columns_)

# generate frequent itemsets using apriori algorithm
```

```
Frequent_itemsets = apriori(df, min_support=0.01, use_colnames=True)

# generate association rules from frequent itemsets
Rules = association_rules(frequent_itemsets, metric=”lift”, min_threshold=1)

# display information
Print(“Original Dataset:\n”)
Print(df.head())
Print(“\nFrequent Itemsets:\n”)
Print(frequent_itemsets)
Print(“\nAssociation Rules:\n”)
Print(rules)
```

@Slip-8

Q. 1) Write a JavaScript to display message ‘Exams are near, have you started preparing for?’ (usealert Box) and Accept any two numbers from user and display addition of two number .(Use Prompt and Confirm box)

AAAns:

```
// Display message using alert box
Alert(‘Exams are near, have you started preparing for?’);

// Accept two numbers from user using prompt and confirm boxes
Let num1 = prompt(‘Enter first number:’);
Let num2 = prompt(‘Enter second number:’);
Let confirmMsg = `Are you sure you want to add ${num1} and ${num2}?`;

// Show confirmation message to user using confirm box
```

```
Let confirmResult = confirm(confirmMsg);


// If user confirms, then perform addition and display the result
If (confirmResult) {
  Num1 = parseInt(num1);
  Num2 = parseInt(num2);
  Let sum = num1 + num2;
  Alert(`The sum of ${num1} and ${num2} is ${sum}.`);
}
```

Q. 2)Download the groceries dataset. Write a python program to read the dataset and display its

Information. Preprocess the data (drop null values etc.) Convert the categorical values into numeric

Format. Apply the apriori algorithm on the above dataset to generate the frequent itemsets and association

Rules.

Ans:

```
Import pandas as pd
From mlxtend.preprocessing import TransactionEncoder
From mlxtend.frequent_patterns import apriori, association_rules


# Load the dataset
Df = pd.read_csv('market_basket.csv')


# Drop any rows with null values
Df.dropna(inplace=True)


# Convert categorical values to numeric format
Te = TransactionEncoder()
```

```
Te_ary = te.fit(df.values).transform(df.values)

Df = pd.DataFrame(te_ary, columns=te.columns_)


# Generate frequent itemsets

Frequent_itemsets = apriori(df, min_support=0.01, use_colnames=True)


# Generate association rules

Rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)


# Display information about the dataset

Print("Dataset information:")

Print(df.info())


# Display the frequent itemsets

Print("\nFrequent itemsets:")

Print(frequent_itemsets)


# Display the association rules

Print("\nAssociation rules:")

Print(rules)
```

@Slip-9

Q. 1) Write a JavaScript function to validate username and password for a membership form.

Ans:

```
Function validateForm() {

  // Get the username and password input values
  Var username = document.forms[“membershipForm”][“username”].value;
  Var password = document.forms[“membershipForm”][“password”].value;

  // Validate username
  If (username == “”) {
    Alert(“Username must be filled out”);
    Return false;
  }

  // Validate password
  If (password == “”) {
    Alert(“Password must be filled out”);
    Return false;
  }

  // Return true if both username and password are valid
  Return true;
}
```

Q. 2)Create your own transactions dataset and apply the above process on your dataset.

Ans:

```
Items=[‘item1’,’item2’,’item3’,’item4’]

Transactions = [    [‘item1’, ‘item2’, ‘item3’],
    [‘item2’, ‘item3’],
    [‘item1’, ‘item2’, ‘item4’],
    [‘item1’, ‘item4’],
    [‘item2’, ‘item3’, ‘item4’],
    [‘item1’, ‘item3’, ‘item4’],
    [‘item1’, ‘item2’],
    [‘item1’, ‘item3’],
    [‘item3’, ‘item4’],
    [‘item2’, ‘item4’]
]

From mlxtend.preprocessing import TransactionEncoder
From mlxtend.frequent_patterns import apriori, association_rules

# Convert the transactions into a binary matrix
Te = TransactionEncoder()
Te_ary = te.fit_transform(transactions)

# Convert the binary matrix into a pandas DataFrame
Df = pd.DataFrame(te_ary, columns=te.columns_)

# Generate frequent itemsets with a minimum support of 0.3
Frequent_itemsets = apriori(df, min_support=0.3, use_colnames=True)

# Generate association rules with a minimum confidence of 0.7
Association_rules = association_rules(frequent_itemsets, metric=”confidence”, min_threshold=0.7)

# Print the frequent itemsets and association rules
Print(frequent_itemsets)
Print(association_rules)
```

@Slip-10

Q. 1) Create a HTML fileto insert text before and after a Paragraph using jQuery. [Hint : Use before( ) And after( )].

Ans:

```
<!DOCTYPE html>
<html>
<head>
        <title>Insert text before and after paragraph using jQuery</title>
        <script src=https://code.jquery.com/jquery-3.6.0.min.js></script>
</head>
<body>
        <h1>Insert text before and after paragraph using jQuery</h1>

        <p>This is a paragraph.</p>

        <script>
                $(document).ready(function() {
                        $("p").before("Text inserted before the paragraph. ");
                        $("p").after(" Text inserted after the paragraph.");
                });
        </script>
</body>
</html>
```

Q2).Create the following dataset in python & Convert the categorical values into numeric format.Apply

The apriori algorithm on the above dataset to generate the frequent itemsets and association rules. Repeat

The process with different min_sup values.

TID={1:[“eggs”,”milk”,”bread”],2=[“eggs”,”apple”],3=[“milk”,”bread”],4=[“apple”,”milk”],5=[“milk”,”apple”,”bread”]}

Ans:

```
Import pandas as pd

From mlxtend.preprocessing import TransactionEncoder

From mlxtend.frequent_patterns import apriori, association_rules


# Create the dataset

Dataset = {

   1: [“eggs”,”milk”,”bread”],

   2: [“eggs”,”apple”],

   3: [“milk”,”bread”],

   4: [“apple”,”milk”],

   5: [“milk”,”apple”,”bread”]

}


# Convert categorical values into numeric format

Te = TransactionEncoder()

Te_ary = te.fit(dataset.values()).transform(dataset.values())

Df = pd.DataFrame(te_ary, columns=te.columns_)
```

```
# Apply Apriori algorithm to generate frequent itemsets and association rules
Min_sup = 0.4
Frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
Association_rules = association_rules(frequent_itemsets, metric=”confidence”, min_threshold=0.6)

# Print the frequent itemsets and association rules
Print(“Frequent Itemsets:\n”, frequent_itemsets)
Print(“\nAssociation Rules:\n”, association_rules)
```

@Slip-11

Q. 1) Write a Javascript program to accept name of student, change font color to red, font size to 18 if Student name is present otherwise on clicking on empty text box display image which changes its size (Use onblur, onload, onmousehover, onmouseclick, onmouseup)

Ans:

```
<!DOCTYPE html>
<html>
<head>
        <title>JavaScript Example</title>
        <style>
                #name {
                        Font-size: 14px;
                        Color: black;
                }
        </style>
</head>
<body>
```

```
        <input type=”text” id=”name” onblur=”changeStyle()” onmouseover=”changeSize()”
onmouseout=”resetSize()” onmousedown=”changeColor()” onmouseup=”resetColor()”>

        <img id=”img” src=https://via.placeholder.com/150 onload=”changeImageSize()”>


        <script>

                Function changeStyle() {

                        Let name = document.getElementById(“name”).value;

                        If (name) {

                                Document.getElementById(“name”).style.fontSize = “18px”;

                                Document.getElementById(“name”).style.color = “red”;

                        } else {

                                Document.getElementById(“img”).style.display = “block”;

                        }

                }


                Function changeSize() {

                        Document.getElementById(“name”).style.fontSize = “16px”;

                }


                Function resetSize() {

                        Document.getElementById(“name”).style.fontSize = “14px”;

                }


                Function changeColor() {

                        Document.getElementById(“name”).style.color = “blue”;

                }


                Function resetColor() {

                        Document.getElementById(“name”).style.color = “red”;
```

```
                    }

                    Function changeImageSize() {
                              Document.getElementById("img").style.width = "200px";
                              Document.getElementById("img").style.height = "200px";
                    }
          </script>
</body>
</html>
```

Q 2).Create the above dataset in python & Convert the categorical values into numeric format.Apply

The apriori algorithm on the above dataset to generate the frequent itemsets and associationrules. Repeat

The process with different min_sup values.

TID={1:["butter","bread","milk],2=["butter","flour","milk","suger"],3=["butter","eggs","milk","salt"],4=["eggs"],5=["butter","flour","milk","salt"]}

Ans:

```
Import pandas as pd
From mlxtend.preprocessing import TransactionEncoder
From mlxtend.frequent_patterns import apriori, association_rules

# Creating the dataset
```

```
Dataset = [['butter', 'bread', 'milk'], ['butter', 'flour', 'milk', 'sugar'], ['butter', 'eggs', 'milk', 'salt'],
['eggs'], ['butter', 'flour', 'milk', 'salt']]

Df = pd.DataFrame(dataset)


# Converting the categorical values into numeric format

Te = TransactionEncoder()

Te_ary = te.fit(dataset).transform(dataset)

Df = pd.DataFrame(te_ary, columns=te.columns_)


# Generating frequent itemsets using Apriori algorithm with different min_sup values

Min_sup_values = [0.4, 0.3, 0.2]

For min_sup in min_sup_values:

   Frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)

   Print("Frequent Itemsets with minimum support of", min_sup)

   Print(frequent_itemsets)


   # Generating association rules

   Rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)

   Print("Association Rules with minimum support of", min_sup)

   Print(rules)
```

@Slip-12

Q. 1)Write AJAX program to read contact.dat file and print the contents of the file in a tabular format

When the user clicks on print button. Contact.dat file should contain srno, name, residence number, Mobile number, Address. [Enter at least 3 record in contact.dat file]

.

Ans:

Html file

```
<<!DOCTYPE html>
<html>
<head>
        <title>Contact List</title>
        <script src=https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js></script>
        <script src="script.js"></script>
</head>
<body>
        <button id="printBtn">Print Contacts</button>
        <br><br>
        <table id="contactTable">
                <thead>
                        <tr>
                                <th>Sr. No.</th>
                                <th>Name</th>
                                <th>Residence Number</th>
                                <th>Mobile Number</th>
                                <th>Address</th>
                        </tr>
                </thead>
                <tbody>
                        <!—Contact list will be displayed here →
                </tbody>
        </table>
</body>
</html>
```

Ajax file

```
$(document).ready(function() {
        // Event listener for print button
        $("#printBtn").click(function() {
                // AJAX request to read contact.dat file
                $.ajax({
                        url: "contact.dat",
                        dataType: "text",
                        success: function(data) {
                                // Split the file contents into lines
                                Var lines = data.split("\n");

                                // Iterate over each line and create a table row
                                Var tableRows = "";
                                For (var i = 0; i < lines.length; i++) {
                                        Var columns = lines[i].split(",");
                                        If (columns.length == 5) { // Only process valid rows
                                                tableRows += "<tr>";
                                                for (var j = 0; j < columns.length; j++) {
                                                        tableRows += "<td>" + columns[j] + "</td>";
                                                }
                                                tableRows += "</tr>";
                                        }
                                }

                                // Add the table rows to the table body
                                $(“#contactTable tbody”).html(tableRows);
                        },
                        Error: function(jqXHR, textStatus, errorThrown) {
                                Alert(“Error: “ + errorThrown);
                        }
                });
        });
});
```

Q. 2)Create ‘heights-and-weights’ Data set . Build a linear regression model by identifying independent And target variable. Split the variables into training and testing sets and print them. Build a simple linear Regression model for predicting purchases.

Ans:

```
Import numpy as np
Import pandas as pd
From sklearn.linear_model import LinearRegression
From sklearn.model_selection import train_test_split

# Create a random dataset with 10 samples
Heights = np.random.normal(170, 10, 10)
Weights = np.random.normal(70, 5, 10)

# Combine the two arrays into a single dataset
```

```
Dataset = pd.DataFrame({'Height': heights, 'Weight': weights})

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(dataset['Height'], dataset['Weight'], test_size=0.2,
random_state=42)

# Create a Linear Regression model and fit it to the training data
Lr_model = LinearRegression()
Lr_model.fit(X_train.values.reshape(-1, 1), y_train)

# Print the model coefficients
Print('Model Coefficients:', lr_model.coef_)

# Predict the weights for the test data and print the predictions
Y_pred = lr_model.predict(X_test.values.reshape(-1, 1))
Print('Predictions:', y_pred)
```

@Slip-13

Q. 1) Write AJAX program where the user is requested to write his or her name in a text box, and the Server keeps sending back responses while the user is typing. If the user name is not entered then the Message displayed will be, "Stranger, please tell me your name!". If the name is Rohit, Virat, Dhoni, Ashwin or Harbhajan , the server responds with "Hello, master !". If the name is anything else, the Message will be ", I don't know you!".

Ans:

Html file

```
<!DOCTYPE html>
<html>
<head>
        <title>AJAX Program</title>
        <script src=https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js></script>
</head>
<body>
        <label for=”name”>Enter your name:</label>
        <input type=”text” id=”name” name=”name”>
        <div id=”response”></div>

        <script src=”ajax.js”></script>
</body>
</html>
```

Ajax file

```
$(document).ready(function() {
        // Attach an event listener to the name input field
        $(‘#name’).on(‘input’, function() {
                // Get the name entered by the user
                Var name = $(this).val();

                // Send an AJAX request to the server
                $.ajax({
                        url: ‘server.php’,
                        type: ‘POST’,
                        data: { name: name },
                        success: function(response) {
                                // Update the response div with the server’s response
                                $(‘#response’).html(response);
                        }
                });
        });
});
```

File name: Server.php

```
<?php

// Get the name entered by the user
$name = $_POST[‘name’];

// Check if the name is empty
If (empty($name)) {
        Echo ‘Stranger, please tell me your name!’;
}
// Check if the name is one of the master names
Else if ($name == ‘Rohit’ || $name == ‘Virat’ || $name == ‘Dhoni’ || $name == ‘Ashwin’ || $name ==
‘Harbhajan’) {
        Echo ‘Hello, master!’;
}
// Otherwise, the server doesn’t know the user
Else {
```

```
            Echo $name . ‘, I don\’t know you!’;
}
```

Q. 2)Download nursery dataset from UCI. Build a linear regression model by identifying independent And target variable. Split the variables into training and testing sets and print them. Build a simple linear Regression model for predicting purchases.

Ans:

```
Import pandas as pd
Import numpy as np
From sklearn.model_selection import train_test_split
From sklearn.linear_model import LinearRegression

# Load the dataset

url = https://archive.ics.uci.edu/ml/machine-learning-databases/nursery/nursery.data
names = [‘parents’, ‘has_nurs’, ‘form’, ‘children’, ‘housing’, ‘finance’, ‘social’, ‘health’, ‘class’]
dataset = pd.read_csv(url, names=names)

# Identify independent and target variables
X = dataset.drop(‘class’, axis=1)
Y = dataset[‘class’]

# Convert categorical variables into numerical variables using one-hot encoding
X = pd.get_dummies(X)

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Build a linear regression model
Model = LinearRegression()
Model.fit(X_train, y_train)

# Print the coefficients of the model
Print("Intercept: ", model.intercept_)
Print("Coefficients: ", model.coef_)

# Predict the target variable for the testing set
Y_pred = model.predict(X_test)

# Evaluate the model using Mean Squared Error (MSE)
Mse = np.mean((y_test – y_pred) ** 2)
Print("MSE: ", mse)
```

@Slip-14

Q. 1) Create TEACHER table as follows TEACHER(tno, tname, qualification, salary). Write Ajax Program to select a teachers name and print the selected teachers details.

AAns:

Js file

```
<!DOCTYPE html>
<html>
<head>
        <title>Teacher Details</title>
        <script src=https://code.jquery.com/jquery-3.6.0.min.js></script>
```

```
</head>
<body>
        <select id="teacher-list">
                <option value="">--Select Teacher--</option>
                <option value="1">John Doe</option>
                <option value="2">Jane Smith</option>
                <option value="3">Bob Johnson</option>
        </select>
        <button id="submit-btn">Get Details</button>
        <div id="details"></div>

        <script>
                $(document).ready(function() {
                        $('#submit-btn').click(function() {
                                Var tno = $('#teacher-list').val();
                                If (tno == '') {
                                        Alert('Please select a teacher.');
                                        Return;
                                }
                                $.ajax({
                                        url: 'teacherdetails.php',
                                        method: 'POST',
                                        data: {tno: tno},
                                        success: function(response) {
                                                $('#details').html(response);
                                        },
                                        Error: function(xhr, status, error) {
                                                Console.log(xhr.responseText);
                                        }
```

```
                                    });
                          });
                });
        </script>
</body>
</html>
```

Php file teacherdetails.php

```
<?php
// Connect to database
$servername = “localhost”;
$username = “username”;
$password = “password”;
$dbname = “database_name”;
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
If (!$conn) {
   Die(“Connection failed: “ . mysqli_connect_error());
}

// Retrieve selected teacher details
If (isset($_POST[‘tno’])) {
        $tno = $_POST[‘tno’];
        $sql = “SELECT * FROM TEACHER WHERE tno = ‘$tno’”;
        $result = mysqli_query($conn, $sql);

        If (mysqli_num_rows($result) > 0) {
```

```
                    $row = mysqli_fetch_assoc($result);

                    Echo “Teacher Name: “ . $row[‘tname’] . “<br>”;

                    Echo “Qualification: “ . $row[‘qualification’] . “<br>”;

                    Echo “Salary: “ . $row[‘salary’] . “<br>”;

          } else {

                    Echo “No data found.”;

          }

}


// Close database connection

Mysqli_close($conn);

?>
```

Q. 2)Create the following dataset in python & Convert the categorical values into numeric format.Apply

The apriori algorithm on the above dataset to generate the frequent itemsets and association rules. Repeat

The process with different min_sup_values.

TID={1:[“apple”,”mango”,”banana”],2=[“mango”,”banana”, “cabbage”,”carrots”],3=[“mango”,”banana”,carrots”],4=[“mango”,”carrots”]}AAAns:

```
From mlxtend.preprocessing import TransactionEncoder

From mlxtend.frequent_patterns import apriori


# Create the dataset

TID = {1:[“apple”,”mango”,”banana”],

    2:[“mango”,”banana”,”cabbage”,”carrots”],
    3:["mango","banana","carrots"],
    4:["mango","carrots"]}

# Convert the categorical values into numeric format
Te = TransactionEncoder()
Te_ary = te.fit([TID[i] for i in TID]).transform([TID[i] for i in TID])
Df = pd.DataFrame(te_ary, columns=te.columns_)

# Apply the apriori algorithm with different min_sup values
Min_sup_values = [0.25, 0.5, 0.75]
For min_sup in min_sup_values:
  Frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
  Print("Frequent itemsets with min_sup =", min_sup)
  Print(frequent_itemsets)
  Print("\n")
```

@Slip-15

Q. 1) Write Ajax program to fetch suggestions when is user is typing in a textbox. (eg like google Suggestions. Hint create array of suggestions and matching string will be displayed).

Ans:

```
<!DOCTYPE html>
<html>
<head>
        <title>AJAX Auto Suggestions Example</title>
        <script>
                Function fetchSuggestions(str) {
                        If (str.length == 0) {
```

```
                        Document.getElementById("suggestions").innerHTML = "";
                        Return;
                }
                Var suggestions = ["apple", "banana", "cherry", "dates", "elderberry", "fig",
"grape", "honeydew", "kiwi", "lemon"];
                Var matches = [];
                For (var i = 0; i < suggestions.length; i++) {
                        If (suggestions[i].toLowerCase().startsWith(str.toLowerCase())) {
                                Matches.push(suggestions[i]);
                        }
                }
                If (matches.length > 0) {
                        Document.getElementById("suggestions").innerHTML =
matches.join("<br>");
                } else {
                        Document.getElementById("suggestions").innerHTML = "No suggestions
found";
                }
        }
    </script>
</head>
<body>
    <input type="text" onkeyup="fetchSuggestions(this.value)">
    <div id="suggestions"></div>
</body>
</html>
```

Q. 2)Create the following dataset in python & Convert the categorical values into numeric format.Apply

The apriori algorithm on the above dataset to generate the frequent itemsets and association rules. Repeat

The process with different min_sup values.

No	Company	model	year
1.	Tata.	Nexon.	2017
2.	MG.	Astor.	2021
3.	Kia.	Seltos.	2019
4.	Hyundai.	Creta.	2015

Ans:

```
Import pandas as pd

# Create the dataset
Data = {‘No’: [1, 2, 3, 4],
    ‘Company’: [‘Tata’, ‘MG’, ‘Kia’, ‘Hyundai’],
    ‘Model’: [‘Nexon’, ‘Astor’, ‘Seltos’, ‘Creta’],
    ‘Year’: [2017, 2021, 2019, 2015]}

Df = pd.DataFrame(data)

# Convert categorical values into numeric format
Df[‘Company’] = pd.Categorical(df[‘Company’])
Df[‘Model’] = pd.Categorical(df[‘Model’])

Df[‘Company’] = df[‘Company’].cat.codes
Df[‘Model’] = df[‘Model’].cat.codes

Print(df)
```

```
From mlxtend.frequent_patterns import apriori
From mlxtend.frequent_patterns import association_rules

# Generate frequent itemsets with min_sup = 0.5
Frequent_itemsets = apriori(df, min_support=0.5, use_colnames=True)
Print(frequent_itemsets)

# Generate association rules with min_threshold = 0.7
Association_rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)
Print(association_rules)
```

@Slip-16

Q. 1) Write Ajax program to get book details from XML file when user select a book name. Create XML File for storing details of book(title, author, year, price).

Ans:

Xml file book_details.xml

```
<books>
        <book>
                <title>The Great Gatsby</title>
                <author>F. Scott Fitzgerald</author>
                <year>1925</year>
                <price>10.99</price>
        </book>
        <book>
                <title>To Kill a Mockingbird</title>
                <author>Harper Lee</author>
                <year>1960</year>
                <price>8.99</price>
        </book>
        <book>
                <title>1984</title>
                <author>George Orwell</author>
                <year>1949</year>
                <price>6.99</price>
        </book>
        <book>
                <title>Pride and Prejudice</title>
                <author>Jane Austen</author>
                <year>1813</year>
                <price>7.99</price>
        </book>
</books>
```

Ajax file

```
<!DOCTYPE html>
<html>
<head>
    <title>Book Details</title>
    <script src=https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js></script>
    <script>
        $(document).ready(function(){
            $(“select”).change(function(){
```

```
                                        Var book = $(this).val();
                                        $.ajax({
                                                url: “book_details.xml”,
                                                dataType: “xml”,
                                                success: function(xml){
                                                        $(xml).find(‘book’).each(function(){
                                                                Var title = $(this).find(‘title’).text();
                                                                If (title == book) {
                                                                        Var author = $(this).find(‘author’).text();
                                                                        Var year = $(this).find(‘year’).text();
                                                                        Var price = $(this).find(‘price’).text();
                                                                        $(“#details”).html(“Author: “ + author +
“<br>Year: “ + year + “<br>Price: “ + price);
                                                                }
                                                        });
                                                }
                                        });
                                });
                        });
                </script>
</head>
<body>
                <select>
                        <option>Select a book</option>
                        <option>The Great Gatsby</option>
                        <option>To Kill a Mockingbird</option>
                        <option>1984</option>
                        <option>Pride and Prejudice</option>
                </select>
```

```
        <div id=”details”></div>
</body>
</html>
```

Q2).Consider any text paragraph. Preprocess the text to remove any special characters and digits.

Generate the summary using extractive summarization pprocess.

Ans:

```
Import re
Import nltk
From nltk.corpus import stopwords
From nltk.tokenize import sent_tokenize, word_tokenize
From heapq import nlargest

# Sample text paragraph you can write any text

Text = “Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human languages, in particular how to program computers to process and analyze large amounts of natural language data. Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural language generation. The history of natural language processing generally started in the 1950s, although work can be found from earlier periods.”

# Remove special characters and digits
Text = re.sub(‘[^a-zA-Z]’, ‘ ‘, text)

# Tokenize the text into sentences
Sentences = sent_tokenize(text)
```

```
# Tokenize each sentence into words and remove stop words
Stop_words = set(stopwords.words('english'))
Words = []
For sentence in sentences:
   Words.extend(word_tokenize(sentence))
Words = [word.lower() for word in words if word.lower() not in stop_words]

# Calculate word frequency
Word_freq = nltk.FreqDist(words)

# Calculate sentence scores based on word frequency
Sentence_scores = {}
For sentence in sentences:
   For word in word_tokenize(sentence.lower()):
      If word in word_freq:
         If len(sentence.split(' ')) < 30:
            If sentence not in sentence_scores:
               Sentence_scores[sentence] = word_freq[word]
            Else:
               Sentence_scores[sentence] += word_freq[word]

# Generate summary by selecting top 3 sentences with highest scores
Summary_sentences = nlargest(3, sentence_scores, key=sentence_scores.get)
Summary = ' '.join(summary_sentences)

Print(summary)
```

@Slip-17

Q. 1) Write a Java Script Program to show Hello Good Morning message onload event using alert box And display the Student registration from.

Ans:

```
<!DOCTYPE html>
<html>
<head>
	<title>Student Registration Form</title>
	<script>
		Window.onload = function() {
			Alert(“Hello Good Morning!”);
		};
	</script>
</head>
<body>
	<h1>Student Registration Form</h1>
	<form>
		<label for=”name”>Name:</label>
		<input type=”text” id=”name” name=”name” required><br><br>
		<label for=”email”>Email:</label>
		<input type=”email” id=”email” name=”email” required><br><br>
		<label for=”phone”>Phone:</label>
		<input type=”tel” id=”phone” name=”phone” required><br><br>
		<label for=”address”>Address:</label>
		<textarea id=”address” name=”address” required></textarea><br><br>
		<input type=”submit” value=”Submit”>
	</form>
</body>
```

</html>

Q. 2)Consider text paragraph.So, keep working. Keep striving. Never give up. Fall down seven times, get Up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than Hardship. So, keep moving, keep growing, keep learning. See you at work.Preprocess the text to remove Any special characters and digits. Generate the summary using extractive summarization process.

Ans:

```
Import re
From nltk.tokenize import sent_tokenize

# Text paragraph
Text = “So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a
greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep
moving, keep growing, keep learning. See you at work.”

# Remove special characters and digits
Text = re.sub(‘[^A-Za-z]+’, ‘ ‘, text)

# Tokenize the sentences
Sentences = sent_tokenize(text)

# Calculate the score of each sentence based on the number of words
# The sentences with more words will have a higher score
Scores = {}
For sentence in sentences:
   Words = sentence.split()
   Score = len(words)
   Scores[sentence] = score

# Sort the sentences based on their scores
Sorted_sentences = sorted(scores.items(), key=lambda x: x[1], reverse=True)

# Extract the top 2 sentences with the highest scores as the summary
Summary_sentences = [sentence[0] for sentence in sorted_sentences[:2]]
Summary = " ".join(summary_sentences)

# Print the summary
Print(summary)
```

@Slip-18

Q. 1) Write a Java Script Program to print Fibonacci numbers on onclick event.

Ans:

```
<!DOCTYPE html>
<html>
<head>
        <title>Fibonacci Numbers</title>
        <script>
                Function generateFibonacci() {
                        // Get the input value from the user
                        Var input = document.getElementById("inputNumber").value;
                        Var output = document.getElementById("output");

                        // Convert the input to a number
```

```
                    Var n = parseInt(input);

                    // Create an array to store the Fibonacci sequence
                    Var fib = [];

                    // Calculate the Fibonacci sequence up to n
                    Fib[0] = 0;
                    Fib[1] = 1;
                    For (var i = 2; i <= n; i++) {
                        Fib[i] = fib[i-1] + fib[i-2];
                    }

                    // Display the Fibonacci sequence
                    Output.innerHTML = "Fibonacci sequence up to " + n + ": " + fib.join(", ");
                }
        </script>
</head>
<body>
        <h1>Fibonacci Numbers</h1>
        <p>Enter a number:</p>
        <input type="text" id="inputNumber">
        <button onclick="generateFibonacci()">Generate Fibonacci</button>
        <p id="output"></p>
</body>
</html>
```

Q. 2)Consider any text paragraph. Remove the stopwords. Tokenize the paragraph to extract words and

Sentences. Calculate the word frequency distribution and plot the frequencies. Plot the wordcloud of the

Txt.

Ans:

```
# Install the libraries
!pip install nltk matplotlib wordcloud

# Import the necessary modules
Import nltk
From nltk.corpus import stopwords
From nltk.tokenize import word_tokenize, sent_tokenize
From nltk.probability import FreqDist
Import matplotlib.pyplot as plt
From wordcloud import WordCloud

# Download the stopwords corpus
Nltk.download('stopwords')

# Define the text paragraph
Text = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tristique ante et velit vestibulum, vel
pharetra orci iaculis. Nullam mattis risus quis augue tincidunt rhoncus. Morbi varius, arcu vitae
scelerisque laoreet, magna est imperdiet quam, sit amet ultrices lectus justo id enim. Sed dictum
suscipit commodo. Sed maximus consequat risus, nec pharetra nibh interdum quis. Etiam eget quam vel
augue dictum dignissim sit amet nec elit. Nunc at sapien dolor. Nulla vitae iaculis lorem. Suspendisse
potenti. Sed non ante turpis. Morbi consectetur, arcu a vestibulum suscipit, mauris eros convallis nibh,
nec feugiat orci enim sit amet enim. Aliquam erat volutpat. Etiam vel nisi id neque viverra dapibus non
non lectus."

# Tokenize the paragraph to extract words and sentences
```

```
Words = word_tokenize(text.lower())

Sentences = sent_tokenize(text)


# Remove the stopwords from the extracted words

Stop_words = set(stopwords.words(‘english’))

Filtered_words = [word for word in words if word.casefold() not in stop_words]


# Calculate the word frequency distribution and plot the frequencies using matplotlib

Fdist = FreqDist(filtered_words)

Fdist.plot(30, cumulative=False)

Plt.show()


# Plot the wordcloud of the text using wordcloud

Wordcloud = WordCloud(width = 800, height = 800,

            Background_color =’white’,

            Stopwords = stop_words,

            Min_font_size = 10).generate(text)


# plot the WordCloud image

Plt.figure(figsize = (8, 8), facecolor = None)

Plt.imshow(wordcloud)

Plt.axis(“off”)

Plt.tight_layout(pad = 0)


Plt.show()
```

@Slip-19

Q. 1) Write a Java Script Program to validate user name and password on onSubmit event.

Ans:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Validate User Name and Password</title>
  <script>
   Function validateForm() {
    Var username = document.forms[“myForm”][“username”].value;
    Var password = document.forms[“myForm”][“password”].value;

    If (username == “”) {
     Alert(“Username must be filled out”);
     Return false;
    }

    If (password == “”) {
     Alert(“Password must be filled out”);
     Return false;
    }
   }
  </script>
 </head>
 <body>
  <h2>Validate User Name and Password</h2>
  <form name=”myForm” onsubmit=”return validateForm()” method=”post”>
   <label for=”username”>Username:</label>
   <input type=”text” id=”username” name=”username”><br><br>
```

```
    <label for=”password”>Password:</label>
    <input type=”password” id=”password” name=”password”><br><br>
    <input type=”submit” value=”Submit”>
  </form>
 </body>
</html>
```

Q. 2)Download the movie_review.csv dataset from Kaggle by using the following link :https://www.kaggle.com/nltkdata/movie-review/version/3?select=movie_review.csv to perform Sentiment analysis on above dataset and create a wordcloud.

Ans:

```
Import pandas as pd
From textblob import TextBlob
From wordcloud import WordCloud, STOPWORDS
Import matplotlib.pyplot as plt

# Load the dataset
Df = pd.read_csv(‘movie_review.csv’)

# Add a column for sentiment analysis using TextBlob
Df[‘Sentiment’] = df[‘Review’].apply(lambda x: TextBlob(x).sentiment.polarity)

# Create a new dataframe for positive reviews only
Pos_df = df[df[‘Sentiment’] > 0.2]

# Create a wordcloud for positive reviews
Wordcloud = WordCloud(width = 800, height = 800,
        Background_color ='white',
        Stopwords = STOPWORDS,
        Min_font_size = 10).generate(' '.join(pos_df['Review']))

# Plot the wordcloud
Plt.figure(figsize = (8, 8), facecolor = None)
Plt.imshow(wordcloud)
Plt.axis("off")
Plt.tight_layout(pad = 0)

Plt.show()
```

@Slip-20

Q. 1) create a student.xml file containing at least 5 student information.

Ans:

```
<?xml version="1.0"?>
<students>
  <student>
    <name>John Doe</name>
    <age>21</age>
    <gender>Male</gender>
    <major>Computer Science</major>
    <gpa>3.8</gpa>
  </student>
  <student>
    <name>Jane Smith</name>
```

```xml
    <age>19</age>
    <gender>Female</gender>
    <major>Business</major>
    <gpa>3.5</gpa>
  </student>
  <student>
    <name>Tom Johnson</name>
    <age>20</age>
    <gender>Male</gender>
    <major>Engineering</major>
    <gpa>3.2</gpa>
  </student>
  <student>
    <name>Sara Lee</name>
    <age>22</age>
    <gender>Female</gender>
    <major>Psychology</major>
    <gpa>3.6</gpa>
  </student>
  <student>
    <name>Mike Brown</name>
    <age>18</age>
    <gender>Male</gender>
    <major>Education</major>
    <gpa>3.4</gpa>
  </student>
</students>
```

Q. 2)Consider text paragraph.”””Hello all, Welcome to Python Programming Academy. Python Programming Academy is a nice platform to learn new programming skills. It is difficult to get enrolled In this Academy.”””Remove the stopwords.

Ans:

```
Import nltk
From nltk.corpus import stopwords
Nltk.download(‘stopwords’)

# Text paragraph
Text = “Hello all, Welcome to Python Programming Academy. Python Programming Academy is a nice
platform to learn new programming skills. It is difficult to get enrolled in this Academy.”

# Tokenize the text
Tokens = nltk.word_tokenize(text)

# Remove stopwords
Stop_words = set(stopwords.words(‘english’))
Filtered_tokens = [word for word in tokens if not word.lower() in stop_words]

# Print the filtered tokens
Print(filtered_tokens)
```

@Slip-21

Q. 1)Add a JavaScript File in Codeigniter. The Javascript code should check whether a number is Positive or negative.

Ans:

Html file

```
<!DOCTYPE html>
<html>
  <head>
    <title>Number Check</title>
    <script src="<?php echo base_url('js/numberCheck.js'); ?>"></script>
  </head>
  <body>
    <h1>Number Check</h1>
    <p>Enter a number to check:</p>
    <input type="number" id="num" />
    <button onclick="checkNumber(document.getElementById('num').value)">Check</button>
  </body>
</html>
```

Create is file check number.js

```
Function checkNumber(num) {
  If (num > 0) {
    Alert("The number is positive.");
  } else if (num < 0) {
    Alert("The number is negative.");
  } else {
    Alert("The number is zero.");
  }
}
```

Q. 2)Build a simple linear regression model for User Data.

Ans:

```
Import pandas as pd

From sklearn.model_selection import train_test_split

From sklearn.linear_model import LinearRegression

From sklearn.metrics import mean_squared_error, r2_score

Import matplotlib.pyplot as plt


# 1. Collect data

Data = pd.read_csv('user_data.csv')


# 2. Preprocess data

Data.dropna(inplace=True)

X = data['age'].values.reshape(-1, 1)

Y = data['income'].values.reshape(-1, 1)


# 3. Split data

X_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)


# 4. Train the model

Regressor = LinearRegression()

Regressor.fit(x_train, y_train)


# 5. Predict values

Y_pred = regressor.predict(x_test)


# 6. Evaluate model

Mse = mean_squared_error(y_test, y_pred)
```

```
R2 = r2_score(y_test, y_pred)
Print("Mean squared error: ", mse)
Print("R-squared: ", r2)

# 7. Visualize results
Plt.scatter(x_test, y_test, color='gray')
Plt.plot(x_test, y_pred, color='red', linewidth=2)
Plt.show()
```

@Slip-22

Q. 1)Create a table student having attributes(rollno, name, class). Using codeigniter, connect to the Database and insert 5 recodes in it.

Ans:

```
<?php

// Establish connection to PostgreSQL database
$conn = pg_connect("host=localhost dbname=your_database_name user=your_username
password=your_password");

// Check if connection was successful
If (!$conn) {
   Echo "Connection failed.";
   Exit;
}
```

```
// Create student table
$query = “CREATE TABLE student (
          Rollno INTEGER PRIMARY KEY,
          Name VARCHAR(50) NOT NULL,
          Class VARCHAR(10) NOT NULL
       )”;
$result = pg_query($conn, $query);

If (!$result) {
   Echo “Error creating table: “ . pg_last_error($conn);
   Exit;
} else {
   Echo “Table created successfully.<br>”;
}

// Insert 5 records into student table
$insert_query = “INSERT INTO student (rollno, name, class)
                 VALUES (1, ‘John Doe’, ‘10A’),
                      (2, ‘Jane Smith’, ‘9B’),
                      (3, ‘Bob Johnson’, ‘11C’),
                      (4, ‘Sarah Lee’, ‘12D’),
                      (5, ‘Tom Brown’, ‘8E’)”;

$insert_result = pg_query($conn, $insert_query);

If (!$insert_result) {
   Echo “Error inserting records: “ . pg_last_error($conn);
   Exit;
} else {
    Echo “Records inserted successfully.”;
}

// Close database connection
Pg_close($conn);

?>
```

Q2).Consider any text paragraph. Remove the stopwords.

Ans:

```
Import nltk
From nltk.corpus import stopwords
From nltk.tokenize import word_tokenize

# sample text paragraph
Text = “Hello all, Welcome to Python Programming Academy. Python Programming Academy is a nice
platform to learn new programming skills. It is difficult to get enrolled in this Academy.”

# tokenize the text paragraph
Words = word_tokenize(text)

# define stopwords
Stop_words = set(stopwords.words(‘english’))

# remove stopwords
Filtered_words = [word for word in words if word.casefold() not in stop_words]
```

```
# join filtered words to form a sentence
Filtered_sentence = ' '.join(filtered_words)

Print(filtered_sentence)
```

@Slip-23

Q. 1) Create a table student having attributes(rollno, name, class) containing atleast 5 recodes . Using Codeigniter, display all its records.

Ans:

```
<?php

// Establish connection to PostgreSQL database
$conn = pg_connect("host=localhost dbname=your_database_name user=your_username
password=your_password");

// Check if connection was successful
If (!$conn) {
   Echo "Connection failed.";
   Exit;
}

// Create student table
$query = "CREATE TABLE student (
          Rollno INTEGER PRIMARY KEY,
          Name VARCHAR(50) NOT NULL,
            Class VARCHAR(10) NOT NULL
        )”;
$result = pg_query($conn, $query);

If (!$result) {
    Echo “Error creating table: “ . pg_last_error($conn);
    Exit;
} else {
    Echo “Table created successfully.<br>”;
}

// Insert 5 records into student table
$insert_query = “INSERT INTO student (rollno, name, class)
                  VALUES (1, ‘John Doe’, ‘10A’),
                         (2, ‘Jane Smith’, ‘9B’),
                         (3, ‘Bob Johnson’, ‘11C’),
                         (4, ‘Sarah Lee’, ‘12D’),
                         (5, ‘Tom Brown’, ‘8E’)”;

$insert_result = pg_query($conn, $insert_query);

If (!$insert_result) {
    Echo “Error inserting records: “ . pg_last_error($conn);
    Exit;
} else {
    Echo “Records inserted successfully.”;
}

// Close database connection
```

```
Pg_close($conn);


// function to display database records
Function display_records($table_name) {
   // establish connection to PostgreSQL database
   $conn = pg_connect("host=localhost dbname=your_database_name user=your_username
password=your_password");

   // check if connection was successful
   If (!$conn) {
      Echo "Connection failed.";
      Exit;
   }

   // retrieve records from specified table
   $query = "SELECT * FROM " . $table_name;
   $result = pg_query($conn, $query);

   // check if query was successful
   If (!$result) {
      Echo "Error retrieving records: " . pg_last_error($conn);
      Exit;
   }

   // display records in an HTML table
   Echo "<table>";
   Echo "<tr><th>Roll No</th><th>Name</th><th>Class</th></tr>";
   While ($row = pg_fetch_assoc($result)) {
        Echo "<tr><td>" . $row['rollno'] . "</td><td>" . $row['name'] . "</td><td>" . $row['class'] .
"</td></tr>";

    }

    Echo "</table>";


    // close database connection

    Pg_close($conn);

}

?>
```

Q2).Consider any text paragraph. Preprocess the text to remove any special characters and

Digits.

Ans:

```
Import re


Text = "Hello, #world123! This is a sample text paragraph. It contains special characters and 5 digits."


# Remove special characters and digits

Processed_text = re.sub(r'[^a-zA-Z\s]', '', text)


Print(processed_text)
```

@Slip-24

Q. 1) Write a PHP script to create student.xml file which contains student roll no, name, address, college And course. Print students detail of specific course in tabular format after accepting course as input.

Ans:

```
<?php
// Define student details
$students = array(
   Array("rollno" => 1, "name" => "John Doe", "address" => "123 Main St", "college" => "ABC College",
"course" => "Computer Science"),
   Array("rollno" => 2, "name" => "Jane Smith", "address" => "456 Main St", "college" => "DEF College",
"course" => "Information Technology"),
   Array("rollno" => 3, "name" => "Bob Johnson", "address" => "789 Main St", "college" => "GHI
College", "course" => "Business Administration"),
   Array("rollno" => 4, "name" => "Sarah Lee", "address" => "101 Main St", "college" => "JKL College",
"course" => "Marketing"),
   Array("rollno" => 5, "name" => "Tom Brown", "address" => "121 Main St", "college" => "MNO
College", "course" => "Computer Science")
);

// Create a SimpleXMLElement object
$xml = new SimpleXMLElement('<students></students>');

// Add student elements to the XML object
Foreach ($students as $student) {
   $student_element = $xml->addChild('student');
   $student_element->addChild('rollno', $student['rollno']);
   $student_element->addChild('name', $student['name']);
   $student_element->addChild('address', $student['address']);
   $student_element->addChild('college', $student['college']);
```

```
    $student_element->addChild(‘course’, $student[‘course’]);
}

// Save the XML data to a file
$xml->asXML(‘student.xml’);

// Get course input from user
$course = isset($_POST[‘course’]) ? $_POST[‘course’] : ‘’;

// Load the XML file
$xml = simplexml_load_file(‘student.xml’);

// Find students with matching course
$filtered_students = $xml->xpath(“//student[course=’$course’]”);

// Print table of matching students
Echo “<table border=’1’>”;
Echo “<tr><th>Roll No</th><th>Name</th><th>Address</th><th>College</th><th>Course</th></tr>”;
Foreach ($filtered_students as $student) {
    Echo “<tr>”;
    Echo “<td>{$student->rollno}</td>”;
    Echo “<td>{$student->name}</td>”;
    Echo “<td>{$student->address}</td>”;
    Echo “<td>{$student->college}</td>”;
    Echo “<td>{$student->course}</td>”;
    Echo “</tr>”;
}
Echo “</table>”;
?>
```

Q. 2) Consider the following dataset :
https://www.kaggle.com/datasets/datasnaek/youtubenew?select=INvideos.csv

Write a Python script for the following :

i.

Read the dataset and perform data cleaning operations on it.

ii.

ii. Find the total views, total likes, total dislikes and comment count.

Ans:

```
Import pandas as pd

# Read the dataset
Df = pd.read_csv(‘INvideos.csv’)

# Drop the columns that are not required
Df = df.drop([‘video_id’, ‘trending_date’, ‘channel_title’, ‘category_id’, ‘publish_time’, ‘tags’,
‘thumbnail_link’, ‘comments_disabled’, ‘ratings_disabled’, ‘video_error_or_removed’], axis=1)

# Convert the datatype of ‘views’, ‘likes’, ‘dislikes’, and ‘comment_count’ to integer
Df[[‘views’, ‘likes’, ‘dislikes’, ‘comment_count’]] = df[[‘views’, ‘likes’, ‘dislikes’,
‘comment_count’]].astype(int)

# Find the total views, likes, dislikes, and comment count
Total_views = df[‘views’].sum()
Total_likes = df[‘likes’].sum()
```

```
Total_dislikes = df['dislikes'].sum()
Total_comments = df['comment_count'].sum()

Print('Total Views:', total_views)
Print('Total Likes:', total_likes)
Print('Total Dislikes:', total_dislikes)
Print('Total Comments:', total_comments)
```

@Slip-25

Q. 1) Write a script to create "cricket.xml" file with multiple elements as shown below:

```
<CricketTeam>
<Team country="Australia">
<player>____</player>
<runs>______</runs>
<wicket>____</wicket>
</Team>
</CricketTeam>
```

Write a script to add multiple elements in "cricket.xml" file of category, country="India".

Ans:

```
<?php
// Create a new DOM document
$doc = new DOMDocument();

// Create the root element
$cricketTeam = $doc->createElement("CricketTeam");
```

```
// Create the first team element for Australia

$teamAustralia = $doc->createElement(“Team”);

$teamAustralia->setAttribute(“country”, “Australia”);


// Create the player element and set its value

$player1 = $doc->createElement(“player”, “Steve Smith”);

$teamAustralia->appendChild($player1);


// Create the runs element and set its value

$runs1 = $doc->createElement(“runs”, “7090”);

$teamAustralia->appendChild($runs1);


// Create the wicket element and set its value

$wicket1 = $doc->createElement(“wicket”, “17”);

$teamAustralia->appendChild($wicket1);


// Append the team element to the root element

$cricketTeam->appendChild($teamAustralia);


// Create the second team element for India

$teamIndia = $doc->createElement(“Team”);

$teamIndia->setAttribute(“country”, “India”);


// Create the player element and set its value

$player2 = $doc->createElement(“player”, “Virat Kohli”);

$teamIndia->appendChild($player2);


// Create the runs element and set its value
```

```
$runs2 = $doc->createElement(“runs”, “12169”);
$teamIndia->appendChild($runs2);

// Create the wicket element and set its value
$wicket2 = $doc->createElement(“wicket”, “4”);
$teamIndia->appendChild($wicket2);

// Create the category element and set its value
$category = $doc->createElement(“category”, “Captain”);
$teamIndia->appendChild($category);

// Append the team element to the root element
$cricketTeam->appendChild($teamIndia);

// Append the root element to the document
$doc->appendChild($cricketTeam);

// Save the XML file
$doc->save(“cricket.xml”);

Echo “Elements added successfully!”;
?>
```

Q. 2) Consider the following dataset : https://www.kaggle.com/datasets/seungguini/youtube-commentsfor-covid19-relatedvideos?select=covid_2021_1.csv

Write a Python script for the following :

i.

Read the dataset and perform data cleaning operations on it.

ii.

ii. Tokenize the comments in words. Iii. Perform sentiment analysis and find the percentage of positive, negative and neutral comments..

Ans:

```
Import pandas as pd
Import nltk
From nltk.sentiment.vader import SentimentIntensityAnalyzer

# read the dataset
Df = pd.read_csv('covid_2021_1.csv')

# remove null values and duplicates
Df.dropna(inplace=True)
Df.drop_duplicates(subset='Comment', inplace=True)

# tokenize comments in words
Nltk.download('punkt')
Df['tokens'] = df['Comment'].apply(nltk.word_tokenize)

# perform sentiment analysis
Nltk.download('vader_lexicon')
Sia = SentimentIntensityAnalyzer()
Df['sentiment'] = df['Comment'].apply(lambda x: sia.polarity_scores(x)['compound'])

# calculate percentage of positive, negative, and neutral comments
```

```
Total_comments = len(df)

Positive_comments = len(df[df['sentiment'] > 0])

Negative_comments = len(df[df['sentiment'] < 0])

Neutral_comments = len(df[df['sentiment'] == 0])

Positive_percentage = (positive_comments / total_comments) * 100

Negative_percentage = (negative_comments / total_comments) * 100

Neutral_percentage = (neutral_comments / total_comments) * 100


# print the results

Print('Total Comments:', total_comments)

Print('Positive Comments:', positive_comments, '(', positive_percentage, '%)')

Print('Negative Comments:', negative_comments, '(', negative_percentage, '%)')

Print('Neutral Comments:', neutral_comments, '(', neutral_percentage, '%)')
```

@Slip-26

Q. 1) Create employee table as follows EMP (eno, ename, designation, salary). Write Ajax program to Select the employees name and print the selected employee's details.

Ans:

Html file

```
<select id="employee-list">
  <option value="">Select an employee</option>
  <!—Populate this dropdown with employee names using PHP →
</select>
```

```
});
```

```
<div id="employee-details">
  <!—Employee details will be displayed here →
</div>
```

Ajax file

```
$(document).ready(function() {
  // Add event listener to the select dropdown
  $('#employee-list').change(function() {
    Var selectedEmployee = $(this).val();
    // Make an AJAX request to fetch employee details
    $.ajax({
      url: 'empdetails.php',
      type: 'POST',
      data: { employeeName: selectedEmployee },
      dataType: 'json',
      success: function(response) {
        // Parse the JSON response and display employee details
        Var detailsHtml = 'Employee Name: ' + response.ename + '<br>' +
                          'Designation: ' + response.designation + '<br>' +
                          'Salary: ' + response.salary;
        $('#employee-details').html(detailsHtml);
      },
      Error: function(xhr, status, error) {
        Console.log('Error:', error);
      }
    });
  });
});
```

Php file as empdetails.php

```
<?php
// Establish database connection
$conn = pg_connect("host=localhost dbname=database_name user=username password=password");
If (!$conn) {
  Die('Connection failed: ' . pg_last_error());
}

// Get the selected employee name from AJAX request
$employeeName = $_POST['employeeName'];

// Query the EMP table for the details of the selected employee
$sql = "SELECT * FROM EMP WHERE ename = '$employeeName'";
$result = pg_query($conn, $sql);

If (pg_num_rows($result) > 0) {
  // Build a JSON object with employee details
  $employee = pg_fetch_assoc($result);
  $response = array(
    'ename' => $employee['ename'],
    'designation' => $employee['designation'],
    'salary' => $employee['salary']
  );
  Echo json_encode($response);
} else {
  Echo "Employee not found";
}

// Close database connection
Pg_close($conn);
?>
```

Q. 2)Consider text paragraph. “””Hello all, Welcome to Python Programming Academy. Python Programming Academy is a nice platform to learn new programming skills. It is difficult to get enrolled In this Academy.””” Preprocess the text to remove any special characters and digits. Generate the Summary using extractive summarization process. Q.

Ans:

```
Import re
From nltk.tokenize import sent_tokenize
From sklearn.feature_extraction.text import TfidfVectorizer
From sklearn.metrics.pairwise import cosine_similarity

# Text to summarize
Text = “Hello all, Welcome to Python Programming Academy. Python Programming Academy is a nice
platform to learn new programming skills. It is difficult to get enrolled in this Academy.”

# Preprocess the text to remove special characters and digits
Preprocessed_text = re.sub(r’[^a-zA-Z\s]’, ‘’, text)

# Tokenize the preprocessed text into sentences
Sentences = sent_tokenize(preprocessed_text)
```

```
# Calculate the importance score of each sentence using TF-IDF
Vectorizer = TfidfVectorizer()
Tfidf_matrix = vectorizer.fit_transform(sentences)
Similarity_matrix = cosine_similarity(tfidf_matrix)

# Select top N sentences based on their importance score
N = 2
Top_sentences = sorted(range(len(similarity_matrix[-1])), key=lambda i: similarity_matrix[-1][i])[-N:]

# Concatenate the top sentences to form the summary
Summary = ‘’
For i in top_sentences:
   Summary += sentences[i] + ‘ ‘

Print(summary)
```

@Slip-27

Q. 1) Create web Application that contains Voters details and check proper validation for (name, Age, and nationality), as Name should be in upper case letters only, Age should not be less than 18 yrs and Nationality should be Indian.(use HTML-AJAX-PHP).

Ans :

Html file

```
<!DOCTYPE html>
<html>
<head>
```

```
        <title>Voter Details</title>
        <script src=https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js></script>
</head>
<body>
        <h2>Voter Details</h2>
        <form id=”voterForm”>
                <label for=”name”>Name:</label>
                <input type=”text” id=”name” name=”name” required><br><br>
                <label for=”age”>Age:</label>
                <input type=”number” id=”age” name=”age” required><br><br>
                <label for=”nationality”>Nationality:</label>
                <input type=”text” id=”nationality” name=”nationality” required><br><br>
                <input type=”submit” value=”Submit”>
        </form>
        <div id=”response”></div>
        <script>
                $(document).ready(function(){
                        $(‘#voterForm’).submit(function(event){
                                Event.preventDefault();
                                Var name = $(‘#name’).val().toUpperCase();
                                Var age = $(‘#age’).val();
                                Var nationality = $(‘#nationality’).val();
                                $.ajax({
                                        url: ‘voter.php’,
                                        method: ‘POST’,
                                        data: {name: name, age: age, nationality: nationality},
                                        success: function(response){
                                                $(‘#response’).html(response);
                                        }
```

```
                              });
                         });
                    });
          </script>
</body>
</html>
```

Voter.php file

```
<?php
$name = $_POST[‘name’];
$age = $_POST[‘age’];
$nationality = $_POST[‘nationality’];

If(preg_match(‘/[^A-Z]/’, $name)){
          Echo ‘Name should be in upper case letters only.’;
} elseif($age < 18) {
          Echo ‘Age should not be less than 18 years.’;
} elseif(strcasecmp($nationality, ‘Indian’) != 0) {
          Echo ‘Nationality should be Indian.’;
} else {
          Echo ‘Validation successful. Voter details: <br>Name: ‘.$name.’<br>Age: ‘.$age.’<br>Nationality: 
‘.$nationality;
}
?>
```

Q. 2) Create your own transactions dataset and apply the above process on your dataset

Ans:

```
Import random

Import csv


# Generate random transaction data

Transactions = []

For i in range(1, 101):

    Transaction_id = i

    Transaction_date = f”2022-05-{random.randint(1, 31):02d}”

    Customer_id = random.randint(1, 10)

    Item_id = random.choice([“A”, “B”, “C”])

    Item_price = round(random.uniform(10.0, 100.0), 2)

    Quantity = random.randint(1, 10)

    Transactions.append([transaction_id, transaction_date, customer_id, item_id, item_price, quantity])


# Save the data to a CSV file

With open(‘transactions.csv’, ‘w’, newline=’’) as csvfile:

    Writer = csv.writer(csvfile)

    Writer.writerow([“Transaction ID”, “Transaction Date”, “Customer ID”, “Item ID”, “Item Price”,
“Quantity”])

    For transaction in transactions:

        Writer.writerow(transaction)


Import pandas as pd


# Read the CSV file into a Pandas DataFrame

Df = pd.read_csv(‘transactions.csv’)


# Convert the “Item Price” column to numeric type
```

```
Df['Item Price'] = pd.to_numeric(df['Item Price'])

# Calculate the sales amount for each transaction
Df['Sales'] = df['Item Price'] * df['Quantity']

# Group the transactions by customer ID and calculate the total sales for each customer
Total_sales = df.groupby('Customer ID')['Sales'].sum().reset_index()

# Print the results
Print(total_sales)
```

@Slip-28

Q. 1) Write a PHP script using AJAX concept, to check user name and password are valid or Invalid (use Database to store user name and password).

Ans:

Html file

```
<!DOCTYPE html>
<html>
<head>
        <title>Login</title>
        <script src=https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js></script>
        <script>
                $(document).ready(function(){
```

```
                              $(“#login-form”).submit(function(event){
                                        Event.preventDefault();
                                        Var username = $(“#username”).val();
                                        Var password = $(“#password”).val();
                                        $.ajax({
                                                  url: ‘check_login.php’,
                                                  type: ‘post’,
                                                  data: {username: username, password: password},
                                                  success: function(response){
                                                            if(response == “valid”){
                                                                      window.location.href = “dashboard.php”;
//redirect to dashboard
                                                            }
                                                            Else{
                                                                      Alert(“Invalid username or password”);
                                                            }
                                                  }
                                        });
                              });
                    });
          </script>
</head>
<body>
          <h2>Login</h2>
          <form id=”login-form” method=”post”>
                    <label>Username:</label>
                    <input type=”text” name=”username” id=”username”><br><br>
                    <label>Password:</label>
                    <input type=”password” name=”password” id=”password”><br><br>
                    <input type=”submit” value=”Login”>
          </form>
</body>
</html>
```

Php file as check_login.php

```
<?php
// Establish database connection
$conn = mysqli_connect(‘localhost’, ‘username’, ‘password’, ‘database_name’);
If (!$conn) {
  Die(‘Connection failed: ‘ . mysqli_connect_error());
}

// Get username and password from AJAX request
$username = $_POST[‘username’];
$password = $_POST[‘password’];

// Query the users table for the entered username and password
$sql = “SELECT * FROM users WHERE username = ‘$username’ AND password = ‘$password’”;
$result = mysqli_query($conn, $sql);

If (mysqli_num_rows($result) > 0) {
  Echo “valid”;
} else {
  Echo “invalid”;
}
```

```
// Close database connection
Mysqli_close($conn);
?>
```

Q. 2) Build a simple linear regression model for Car Dataset.

Ans:

```
From sklearn.linear_model import LinearRegression

Mileage = [[10], [20], [30], [40], [50], [60], [70], [80]]
Price = [24, 19, 17, 13, 10, 7, 5, 2]

Reg = LinearRegression().fit(mileage, price)

Print('Intercept:', reg.intercept_)
Print('Coefficient:', reg.coef_[0])

New_mileage = [[25], [45], [65]]
Predicted_price = reg.predict(new_mileage)

Print('Predicted prices:', predicted_price)
```

@Slip-29

Q. 1) Write a PHP script for the following: Design a form to accept a number from the user.

Perform the operations and show the results.

1) Fibonacci Series.

2) To find sum of the digits of that number.

(Use the concept of self processing page.)

Ans:

```
<!DOCTYPE html>
<html>
<head>
        <title>Number Operations</title>
</head>
<body>
        <h1>Number Operations</h1>
        <?php
        // define variables and set to empty values
        $num = $op = “”;

        If ($_SERVER[“REQUEST_METHOD”] == “POST”) {
                $num = test_input($_POST[“num”]);
                $op = test_input($_POST[“op”]);

                // perform operation based on user’s choice
                Switch ($op) {
                        Case “fib”:
                                $result = fibonacci($num);
                                Echo “<p>The Fibonacci series of $num numbers is: $result</p>”;
                                Break;
                        Case “sum”:
                                $result = sumOfDigits($num);
                                Echo “<p>The sum of digits in $num is: $result</p>”;
                                Break;
                        Default:
                                Echo “<p>Invalid operation selected</p>”;
                }
        }

        Function test_input($data) {
                $data = trim($data);
                $data = stripslashes($data);
                $data = htmlspecialchars($data);
                Return $data;
        }

        Function fibonacci($num) {
                $first = 0;
                $second = 1;
                $result = “”;

                For ($i = 0; $i < $num; $i++) {
                        $result .= $first . “ “;
                        $third = $first + $second;
                        $first = $second;
                        $second = $third;
                }

                Return $result;
        }
```

```
        Function sumOfDigits($num) {
                $sum = 0;

                While ($num > 0) {
                        $digit = $num % 10;
                        $sum += $digit;
                        $num = (int)($num / 10);
                }

                Return $sum;
        }
        ?>

        <form method=”post” action=”<?php echo htmlspecialchars($_SERVER[“PHP_SELF”]);?>”>
                <label for=”num”>Enter a number:</label>
                <input type=”number” name=”num” id=”num” required>
                <br><br>
                <label for=”op”>Select an operation:</label>
                <select name=”op” id=”op” required>
                        <option value=””>--Select--</option>
                        <option value=”fib”>Fibonacci Series</option>
                        <option value=”sum”>Sum of Digits</option>
                </select>
                <br><br>
                <input type=”submit” value=”Submit”>
        </form>
</body>
</html>
```

Q. 2) Build a logistic regression model for Student Score Dataset.

Ans:

```
# Import necessary libraries
Import pandas as pd
From sklearn.linear_model import LogisticRegression
From sklearn.model_selection import train_test_split
From sklearn.metrics import accuracy_score

# Load the dataset
Data = pd.read_csv('student_scores.csv')

# Split the data into input and output variables
X = data.iloc[:, :-1].values
Y = data.iloc[:, -1].values

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Create the logistic regression model and fit it to the training data
Classifier = LogisticRegression()
Classifier.fit(X_train, y_train)

# Make predictions on the testing set
Y_pred = classifier.predict(X_test)

# Evaluate the model's accuracy
```

```
Accuracy = accuracy_score(y_test, y_pred)
Print("Accuracy:", accuracy)
```

@Slip-30

Q. 1) Create a XML file which gives details of books available in "Bookstore" from following

Categories.

1) Yoga

2) Story

3) Technical

And elements in each category are in the following format

```
<Book>
<Book_Title>
--------------</Book_Title>
<Book_Author> ----------------</Book_Author>
<Book_Price>
--------------</Book_Price>
</Book>
```

Save the file as "Bookcategory.xml"

.

Ans:

```
<?xml ve<?xml version="1.0" encoding="UTF-8"?>
<Bookstore>
  <Yoga>
    <Book>
```

```xml
    <Book_Title>Light on Yoga</Book_Title>
    <Book_Author>B.K.S. Iyengar</Book_Author>
    <Book_Price>20.99</Book_Price>
  </Book>
  <Book>
    <Book_Title>The Yoga Bible</Book_Title>
    <Book_Author>Christina Brown</Book_Author>
    <Book_Price>15.50</Book_Price>
  </Book>
</Yoga>
<Story>
  <Book>
    <Book_Title>The Alchemist</Book_Title>
    <Book_Author>Paulo Coelho</Book_Author>
    <Book_Price>12.99</Book_Price>
  </Book>
  <Book>
    <Book_Title>The Da Vinci Code</Book_Title>
    <Book_Author>Dan Brown</Book_Author>
    <Book_Price>14.75</Book_Price>
  </Book>
</Story>
<Technical>
  <Book>
    <Book_Title>Python for Data Science Handbook</Book_Title>
    <Book_Author>Jake VanderPlas</Book_Author>
    <Book_Price>28.99</Book_Price>
  </Book>
  <Book>
```

```
      <Book_Title>Cracking the Coding Interview</Book_Title>

      <Book_Author>Gayle Laakmann McDowell</Book_Author>

      <Book_Price>23.50</Book_Price>

    </Book>

  </Technical>

</Bookstore>
```

Q. 2) Create the dataset . transactions = [[‘eggs’, ‘milk’,’bread’], [‘eggs’, ‘apple’], [‘milk’, ‘bread’], [‘apple’,

‘milk’], [‘milk’, ‘apple’, ‘bread’]] .

Convert the categorical values into numeric format.Apply the apriori algorithm on the above dataset to

Generate the frequent itemsets and association rules.

Ans:

```
Transactions = [[‘eggs’, ‘milk’, ‘bread’], [‘eggs’, ‘apple’], [‘milk’, ‘bread’], [‘apple’, ‘milk’], [‘milk’, ‘apple’, ‘bread’]]

# Create a dictionary to map items to unique numeric values
Item_to_num = {‘eggs’: 1, ‘milk’: 2, ‘bread’: 3, ‘apple’: 4}

# Convert the categorical values in the dataset to numeric values
Numeric_transactions = []
For transaction in transactions:
   Numeric_transaction = [item_to_num[item] for item in transaction]
   Numeric_transactions.append(numeric_transaction)
```

```
Print(numeric_transactions)


From mlxtend.frequent_patterns import apriori, association_rules


# Generate frequent itemsets with a minimum support of 0.4
Frequent_itemsets = apriori(numeric_transactions, min_support=0.4, use_colnames=True)


# Generate association rules with a minimum confidence of 0.7
Rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)


Print(frequent_itemsets)
Print(rules)
```