MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Syllabus and Course Structure

I and ML Lab SEMESTER – VI			
Course Code:	**21BTCS012**	**Course Credits:**	**1**
Teaching Hours / Week (L: T: P):	0:0:2	**CA Marks:**	25
Total Number of Teaching Hours:	**15**	**END-SEM Marks:**	**30**

Prerequisites: Basic knowledge of programming

Companion Course: Programming in Python

Course Objectives:

1. To introduce basic machine learning techniques.
2. To develop the skills in using recent machine learning software for solving practical problems in high-performance computing environment.
3. To develop the skills in applying appropriate supervised, semi-supervised or un-supervised learning algorithms for solving practical problems.
4. Identify innovative research directions in Artificial Intelligence, Machine Learning and BigData analytics

Course Outcomes:
At the end of the course the student will be able to

CO1: Demonstrate the ability to solve problems collaboratively.
CO2: Demonstrate knowledge of artificial intelligence concepts.
CO3: Understand fundamental concepts and methods of machine learning.
CO4: Analyse and evaluate simple algorithms for classification.
CO5: Design simple algorithms for pattern classification, code them with Python programming language and test them with benchmark data sets.
CO6: Practically establish, refine and implement strategies to take the idea in to students 'fraternity.

Learning Resources

Text Books:

1. Buyya, Rajkumar, James Broberg, and Andrzej M. Goscinski, eds. Cloud computing: Principles and paradigms. John Wiley & Sons, 2010.
2. Dan C. Marinescu, "Cloud Computing - Theory and Practice", 1st Edition, Morgan Kaufmann is an imprint of Elsevier, 2013.
3. AWS Whitepapers.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Index

Sr. No.	Name of Experiment/Assignment	Date of Assignment	Date of Submission	Marks out of 10	Dated Signature of Faculty
1	Write a Program to Implement Breadth First Search.				
2	Write a program to implement A* Algorithm				
3	Write a program to implement Tic-Tac-Toe game				
4	a. Implementation of Python basic libraries such as Math, NumPy and Scipy b. Implementation of Python Libraries for ML application such as Pandas and Matplotlib				
5	Installation and configuration of machine learning environment with Anaconda (Jupyter notebook)				
6	Download any dataset from UCI or Data.org or from any data repositories and perform the basic data pre-processing steps using Python/R.				
7	Develop a Bayesian Classifier for IRIS Dataset.				
8	Implement K means algorithm for multidimensional data for Cars and Wine dataset from UCI repository.				

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

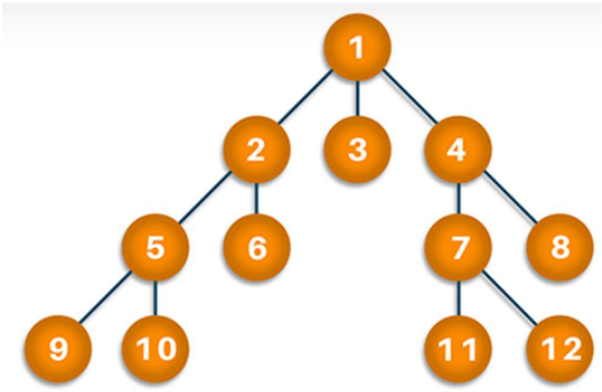
Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 01

Title: Write a Program to Implement Breadth First Search.

Theory:

Breadth-first search and Depth-first search in python are algorithms used to traverse a graph or a tree. They are two of the most important topics that any new python programmer should definitely learn about. Here we will study what breadth-first search in python is, understand how it works with its algorithm, implementation with python code, and the corresponding output to it. Also, we will find out the application and uses of breadth-first search in the real world.



Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph. Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. BFS in python can be implemented by using data structures like a dictionary and lists. Breadth-First Search in tree and graph is almost the same. The only difference is that the graph may contain cycles, so we may traverse to the same node again.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

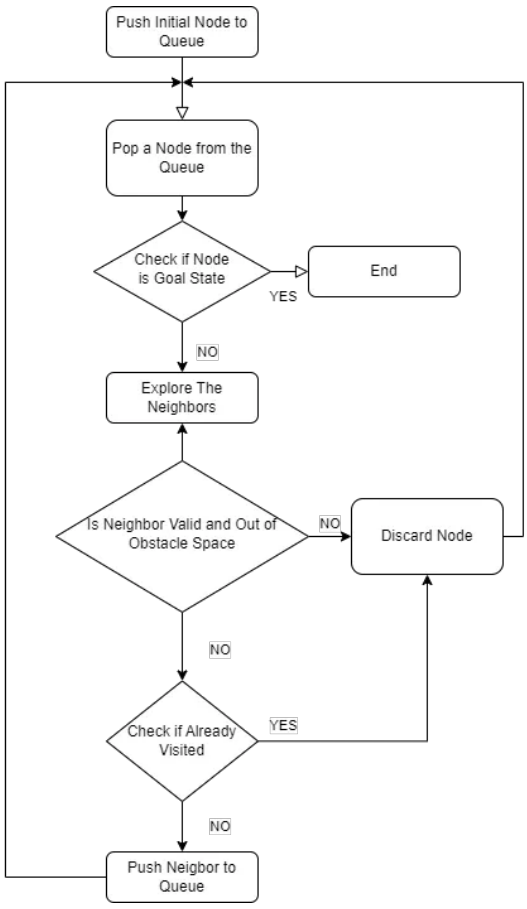
Artificial Intelligence and Machine Learning Lab

Algorithm:

The steps of the algorithm work as follow:

1. Start by putting any one of the graph's vertices at the back of the queue.
2. Now take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add those which are not within the visited list to the rear of the queue.
4. Keep continuing steps two and three till the queue is empty.

Flowchart:



Pseudo code or Program

```
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : ['8'],
  '8' : []
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

```
def rudy_bfs(graph, root):
    visited = []
    queue = [root]

    while queue:
        vertex = queue.pop(0)
        print(vertex, end=" ")

        for neighbour in graph[vertex]:
            if neighbour not in visited:
                visited.append(neighbour)
                queue.append(neighbour)



rudy_bfs(graph, '5')
```

Outputs

```
In [9]: rudy_bfs(graph, '5')
5 3 7 2 4 8
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Exercise for Practice	
Q1.	How Breadth First Search algorithm works?
Ans:	Breadth-First Search (BFS) starts at a given node, then explores all its neighbors before moving on to their unvisited neighbors. This process continues until all nodes in the graph have been visited, ensuring a level-by-level traversal.
Q2.	What are the applications of Breadth First Search algorithm in real world?
Ans	BFS is widely used in network routing protocols to find all reachable nodes. also used in social platforms to find people within a certain dist level and in web crawlers to visit web pages efficiently
Conclusion	In conclusion, implementing the BFS algorithm is a valuable exercise in understanding graph traversal techniques. It provides hands-on experience with data structures like queues and sets, and concepts like node visitation and level-by-level exploration

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 02

Title: Write a program to implement A* Algorithm

Theory:

A* search algorithm is an algorithm that separates it from other traversal techniques. This makes A* smart and pushes it much ahead of conventional algorithms.

Let's try to understand Basic AI Concepts and comprehend how does A* algorithm work. Imagine a huge maze that is too big that it takes hours to reach the endpoint manually. Once you complete it on foot, you need to go for another one. This implies that you would end up investing a lot of time and effort to find the possible paths in this maze. Now, you want to make it less time-consuming. To make it easier, we will consider this maze as a search problem and will try to apply it to other possible mazes we might encounter in due course, provided they follow the same structure and rules.

As the first step to converting this maze into a search problem, we need to define these six things.

1. A set of prospective states we might be in
2. A beginning and end state
3. A way to decide if we've reached the endpoint
4. A set of actions in case of possible direction/path changes
5. A function that advises us about the result of an action
6. A set of costs incurring in different states/paths of movement

A* Search Algorithm Steps

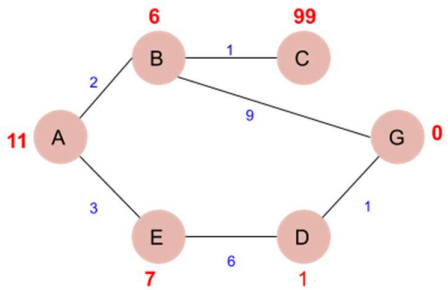
Step 1: Add the beginning node to the open list
Step 2: Repeat the following step

In the open list, find the square with the lowest F cost, which denotes the current square. Now we move to the closed square.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

	Consider 8 squares adjacent to the current square and Ignore it if it is on the closed list or if it is not workable. Do the following if it is workable. Check if it is on the open list; if not, add it. You need to make the current square as this square's a parent. You will now record the different costs of the square, like the F, G, and H costs. If it is on the open list, use G cost to measure the better path. The lower the G cost, the better the path. If this path is better, make the current square as the parent square. Now you need to recalculate the other scores – the G and F scores of this square. – **You'll stop:** If you find the path, you need to check the closed list and add the target square to it. There is no path if the open list is empty and you cannot find the target square. **Step 3.** Now you can save the path and work backward, starting from the target square, going to the parent square from each square you go, till it takes you to the starting square. You've found your path now.
Flowchart: **(if any)**	In this section, we are going to find out how the A* search algorithm can be used to find the most cost-effective path in a graph. Consider the following graph below.  The numbers written on edges represent the distance between the nodes, while the numbers written on nodes represent the heuristic values. Let us

find the most cost-effective path to reach from start state A to final state G using the A* Algorithm.

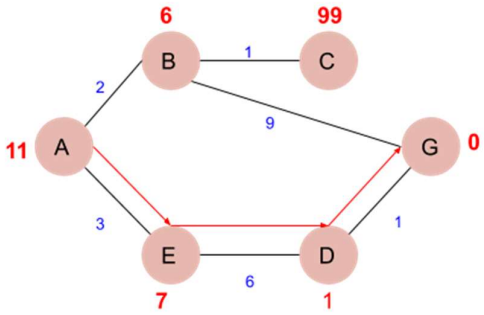
Let's start with node A. Since A is a starting node, therefore, the value of g(x) for A is zero, and from the graph, we get the heuristic value of A is 11, therefore

```
g(x) + h(x) = f(x)

0+ 11 =11

Thus for A, we can write

A=11

Now from A, we can go to point B or point E, so we compute
f(x) for each of them

A → B = 2 + 6 = 8

A → E = 3 + 6 = 9

Since the cost for  A → B is less, we move forward with this
path and compute the f(x) for the children nodes of B

Since there is no path between C and G, the heuristic cost is
set to infinity or a very high value

A → B → C = (2 + 1) + 99= 102

A → B → G = (2 + 9 ) + 0 = 11

Here the path A → B → G has the least cost but it is still
more than the cost of A → E, thus we explore this path
further

A → E → D = (3 + 6) + 1 = 10

Comparing the cost of A → E → D with all the paths we got so
far and as this cost is least of all we move forward with
this path. And compute the f(x) for the children of D

A → E → D → G = (3 + 6 + 1) +0 =10

Now comparing all the paths that lead us to the goal, we
conclude that A → E → D → G is the most cost-effective path
to get from A to G.
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab



Next, we write a program in Python that can find the most cost-effective path by using the a-star algorithm.

First, we create two sets, viz- open and close. The open contains the nodes that have been visited, but their neighbours are yet to be explored. On the other hand, close contains nodes that, along with their neighbours, have been visited.

Pseudo code or Program

```
from queue import PriorityQueue

# Define the A* algorithm function
def a_star_algorithm(graph, start, goal, h):
    # Initialize the open set with the start node
    open_set = PriorityQueue()
    open_set.put((h[start], start))

    # Initialize the came_from dictionary
    #to reconstruct the path later
    came_from = {}

    # Initialize g_score and f_score
    #dictionaries with infinite values

    g_score = {node: float('inf') for node in graph}
    f_score = g_score.copy()

    # Set the g_score and f_score for the start node
    g_score[start] = 0
    f_score[start] = h[start]
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

Artificial Intelligence and Machine Learning Lab

```
# Loop until there are no nodes left to explore
while not open_set.empty():
    # Get the node with the lowest
    #f_score from the open set
    current = open_set.get()[1]

    # If the goal is reached, reconstruct
    #and return the path
    if current == goal:
        return reconstruct_path(came_from,
                                start, goal)

    # Explore neighbors of the current node
    for neighbor, weight in graph.get(current, []):
        # Calculate tentative g_score
        #for the neighbor
        tentative_g_score = g_score[current] + weight

        # If the tentative g_score is
        #better, update path and scores
        if tentative_g_score < g_score[neighbor]:
            came_from[neighbor] = current
            g_score[neighbor] = tentative_g_score
            f_score[neighbor] = tentative_g_score
                                + h[neighbor]
           open_set.put((f_score[neighbor], neighbor))

# If the goal was not reached, return an empty path
return []
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

Artificial Intelligence and Machine Learning Lab

```
# Function to reconstruct the path from start to goal
def reconstruct_path(came_from, start, goal):
    path = [goal]
    while goal in came_from:
        goal = came_from[goal]
        path.append(goal)
    path.reverse()
    return path


Graph_nodes = {
    'A': [('B', 6), ('F', 3)],
    'B': [('C', 3), ('D', 2)],
    'C': [('D', 1), ('E', 5)],
    'D': [('E', 8)],
    'E': [('I', 5), ('J', 5)],
    'F': [('G', 1), ('H', 7)],
    'G': [('I', 3)],
    'H': [('I', 2)],
    'I': [('J', 3)],
    'J': []
}

H_dist = {
    'A': 10,
    'B': 8,
    'C': 5,
    'D': 7,
    'E': 3,
    'F': 6,
    'G': 5,
    'H': 3,
    'I': 1,
    'J': 0
}



# Run the A* algorithm and print the path
start_node = 'A'
goal_node = 'J'
path = a_star_algorithm(Graph_nodes,
                        start_node, goal_node, H_dist)

print(f"Path from {start_node} to
                  {goal_node}: {path}")
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Outputs	`Path from A to J: ['A', 'F', 'G', 'I', 'J']` (Students can attach extra pages if necessary)

Exercise for Practice

Q1.	How does the A * algorithm work?
Ans:	It combines elements of Dijkstra's shortest path algorithm and a heuristic to find the shortest path between two nodes in a graph. By selecting nodes with the lowest total cost ('f = g + h').
Q2.	Why is the A* algorithm popular?
Ans	A* is favored due to its efficiency and ability to find optimal paths. Unlike other traversal techniques, it incorporates a smart heuristic, Its balance between accuracy and speed makes it widely used.
Q3	Why A* better than Dijkstra?
Ans	because it considers both actual cost and estimated cost (heuristic) during traversal. Dijkstra only considers actual costs. A* is more efficient when searching for paths, especially in scenarios with obstacles.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015
(Maharashtra Act No. XXXIX of 2015)

Artificial Intelligence and Machine Learning Lab

Conclusion	In summary, the A algorithm* strikes a balance between efficiency and optimality by incorporating both actual cost and heuristic estimates. Its ability to find optimal paths in graphs, such as maps & games,

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

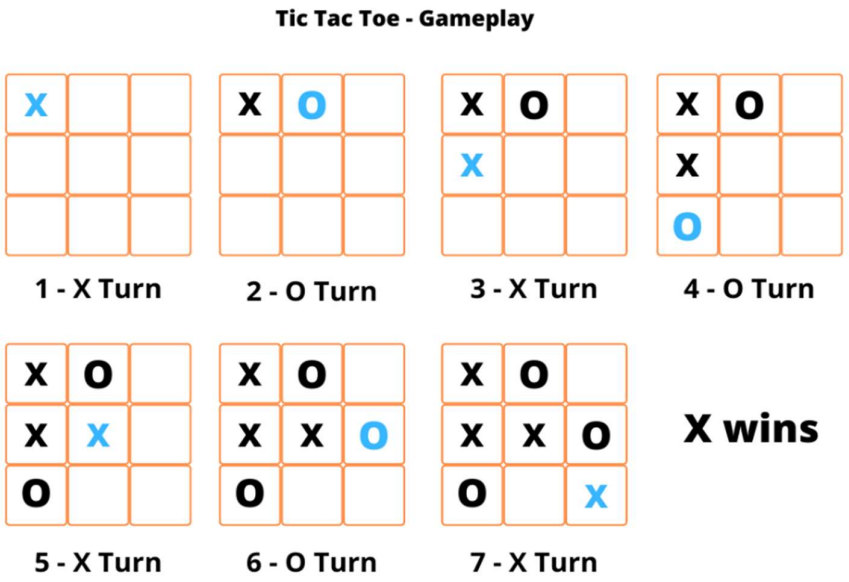
Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 03

Title: Write a program to implement Tic-Tac-Toe game

Theory:

Gaming is one of the entertainments that humans have. We can find different types of games on the web, mobile, desktop, etc. We are not here to make one of those heavy games now. We are going to create a CLI tic-tac-toe game using python.



Flowchart:

(if any)

Algorithm

Create a board using a 2-dimensional array and initialize each element as empty.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

- Using hyphen here to represent empty space '-'.

- Write a function to check whether the board is filled or not.
- Iterate over the board and return false if the board contains an empty sign or else return true.
- Write a function to check whether a player has won or not.
- We have to check all the possibilities that we discussed in the previous section.
- Check for all the rows, columns, and two diagonals.
- Write a function to show the board as we will show the board multiple times to the users while they are playing.
- Write a function to start the game.
- Select the first turn of the player randomly.
- Write an infinite loop that breaks when the game is over (either win or draw).
- Show the board to the user to select the spot for the next move.
- Ask the user to enter the row and column number.
- Update the spot with the respective player sign.
- Check whether the current player won the game or not.
- If the current player won the game, then print a winning message and break the infinite loop.
- Next, check whether the board is filled or not.
- If the board is filled, then print the draw message and break the infinite loop.
- Finally, show the user the final view of the board.

Pseudo code or Program

```
def make_board():
    board = []
    for i in range(3):
        board.append(['-'] * 3)
    return board
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

Artificial Intelligence and Machine Learning Lab

```
def display_board(board):
    for row in board:
        print(' '.join(row))
    print()


def place_player(board, row,
                 col, player):
    board[row][col] = player



def if_won(board, player):
    return any(
        all(board[i][j] == player for
                          j in range(3)) or
        all(board[j][i] == player for
                          j in range(3)) or
        all(board[i][i] == player for
                          i in range(3)) or
        all(board[i][2 - i] == player
                          for i in range(3))
        for i in range(3)
    )
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

```
def start():
    board = make_board()
    player = 'X'

    while True:
        display_board(board)

        row, col = map(
              int, input(f"Player {player},
              enter row and column numbers
              to place :: ").split())
        place_player(board, row - 1,
                     col - 1, player)

        if if_won(board, player):
            print(f"Player {player}
                          wins the game!")
            break

        player = 'X' if player == 'O'
                                  else 'O'

    display_board(board)
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Outputs

```
In [43]: start()

- - -
- - -
- - -

Player X, enter row and column numbers to place :: 1 1
X - -
- - -
- - -

Player O, enter row and column numbers to place :: 2 3
X - -
- - O
- - -

Player X, enter row and column numbers to place :: 2 1
X - -
X - O
- - -

Player O, enter row and column numbers to place :: 3 2
X - -
X - O
- O -

Player X, enter row and column numbers to place :: 3 1
Player X wins the game!
X - -
X - O
X O -
```

Exercise for Practice

Q1. What are the rules of the game?

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Ans:	Tic Tac Toe is a two-player game played on a 3x3 grid. Players take turns marking a square with their symbol (X or O). The first player to align three of symbols vertically, horizontally, diagonally wins.
Q2.	What is the purpose of playing tic-tac-toe?
Ans	Its purpose is to provide a simple, competitive game where players can develop their strategic thinking and problem-solving skills.
Q3	How is tic-tac-toe educational?
Ans	Its educational as it helps players develop strategic thinking pattern recognition. and planning skills. It teaches the concept of actions and consequences
Conclusion	In conclusion, coding a Tic Tac Toe game is an excellent exercise in understanding the fundamentals of programming such as loops, conditionals, and data structures. Allowing to learn game logic, UI, while improving critical thinking

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 04

Title:
a. Implementation of Python basic libraries such as Math, NumPy and Scipy
b. Implementation of Python Libraries for ML application such as Pandas and Matplotlib

Theory:

Installation Of NumPy Command –

pip3 install NumPy To check the version

Import NumPy as np

Installation of Scipy-stack Command –

pip3 install scipy-stack

NUMPY/SCIPY:-

NumPy is a Python library, which stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object, provide tools for integrating C, C++ etc.

NumPy array can also be used as an efficient multi-dimensional container for generic .data.

The nparray (NumPy Array) is a multidimensional array used to store values of same datatype. These arrays are indexed just like Sequences, starts with zero.

The ndarrays are better than regular arrays in terms of faster computation and ease of manipulation.

In different algorithms of Machine Learning like K-means Clustering, Random Forest etc. we have to store the values in an array. So, instead of using regular array, ndarray helps us to manipulate and execute easily.

Installation of scikit

Artificial Intelligence and Machine Learning Lab

Command pip3 install -u scikit-learn

SCIKIT:

- The functionality that scikit-learn provides include:
- Regression, including Linear and Logistic Regression
- Classification, including K-Nearest Neighbours
- Clustering, including K-Means and K-Means++
- Model selection
- Preprocessing, including Min-Max Normalization.

Installation of Pandas Command- pip3 install pandas

PANDAS:

- Merging and Joining Data Sets.
- Reshaping & pivoting Data Sets.
- Inserting & deleting columns in Data Structure.
- Aligning data & dealing with missing data.
- Iterating over a Data set.
- Analysing Time Series.
- Filtering Data around a condition.
- Arranging Data in an ascending & descending.
- Reading from flies with CSV, TXT, XLSX, other formats.
- Manipulating Data using integrated indexing for DataFrame objects.
- Generating Data range, date shifting, lagging, converting frequency, and other other Time Series functionality.
- Sub-setting fancy indexing, & label-based slicing Data Sets that are large in size.
- Performing split apply combine on Data Sets using the group by engine. With Python Pandas, it is easier to clean & wrangle with your Data. features of Pandas make it a great choice for Data Science and Analysis.

Installation of Matplotlib Command- pip3 install matplotlib

MATPLOTLIB:

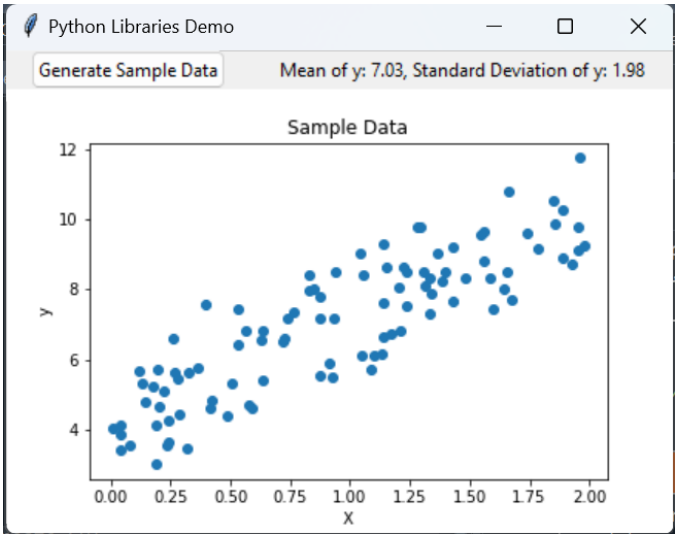
- Matplotlib is a visualization library in Python for 2D plots of arrays. It consists of several plots like line, bar, scatter, histogram etc.
- Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.It can also be used with graphics toolkits like PyQt and wxPython.

```
import tkinter as tk
from tkinter import ttk
import numpy as np
from scipy.stats import norm
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

def generate_sample_data():
    np.random.seed(0)
    X = 2 * np.random.rand(100, 1)
    y = 4 + 3 * X + np.random.randn(100, 1)
    return X, y

def fit_normal_distribution(y):
    mu, std = norm.fit(y.flatten())
    return mu, std

def visualize_data(X, y):
    fig, ax = plt.subplots()
    ax.scatter(X, y)
    ax.set_xlabel('X')
    ax.set_ylabel('y')
    ax.set_title('Sample Data')
    return fig

def on_generate_data():
    X, y = generate_sample_data()
    mu, std = fit_normal_distribution(y)
    fig = visualize_data(X, y)

    canvas = FigureCanvasTkAgg(fig, master=root)
    canvas_widget = canvas.get_tk_widget()
    canvas_widget.grid(row=1, column=0, columnspan=2)

    result_label.config(text=f"Mean of y: {mu:.2f},
                              Standard Deviation of y: {std:.2f}")

# Create GUI
root = tk.Tk()
root.title("Python Libraries Demo")

# Button to generate sample data
generate_button = ttk.Button(root, text="Generate Sample Data",
                                    command=on_generate_data)
generate_button.grid(row=0, column=0)

# Label to display result
result_label = ttk.Label(root, text="")
result_label.grid(row=0, column=1)

root.mainloop()
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab



Output:

(Students can attach extra pages if necessary)

Exercise for Practice

Q1. What are the uses of NumPy?

Ans: NumPy is used for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

Q2. What are the uses of SciPy?

Ans SciPy builds on NumPy and provides a wide range of scientific computing tools. It includes modules for optimization, integration, interpolation, linear algebra, signal processing, and more.

Q3 What are the benefits of Pandas?

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Ans	Pandas simplifies data manipulation and analysis in Python. Its DataFrame object allows for easy handling of structured data, including tasks like data cleaning, merging, reshaping, and analysis.
Q4.	What are the features of Pandas?
Ans	Pandas offers powerful features such as data alignment, missing data handling, group-by operations, time-series functionality, and integration with databases and other data formats like CSV and Excel.
Q5	What is the primary purpose of Matplotlib in Python?
Ans	Matplotlib is primarily used for creating visualizations in Python. It provides a wide variety of plotting options, including line plots, bar plots, scatter plots, histograms, and more, making it suitable for data exploration and presentation.
Conclusion	Thus, we have described the libraries and the installation of the libraries

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 05

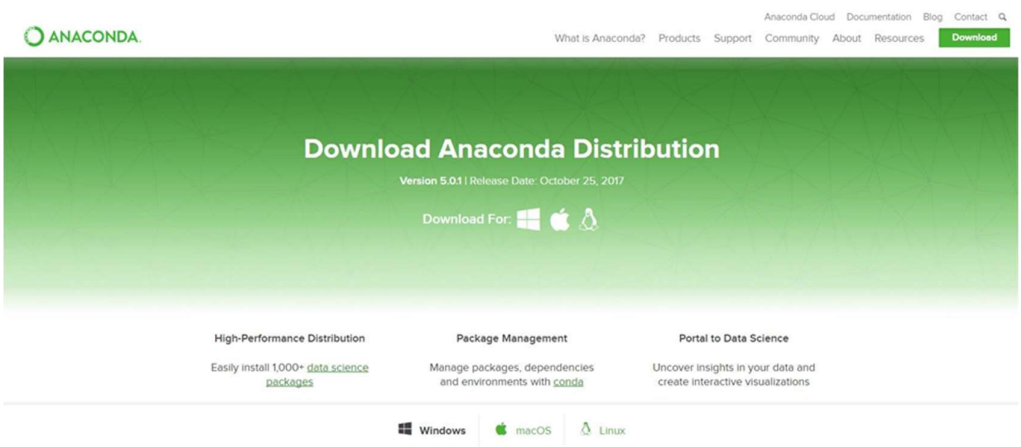
Title: Installation and Configuration of machine learning environment with Anaconda on windows or Ubuntu (Jupyter notebook)

Theory:

Installation Of Anaconda in Windows

1. Visit the Anaconda downloads page

Go to the following link: Anaconda.com/downloads



2. Select Windows

Select Windows where the three operating systems are listed.

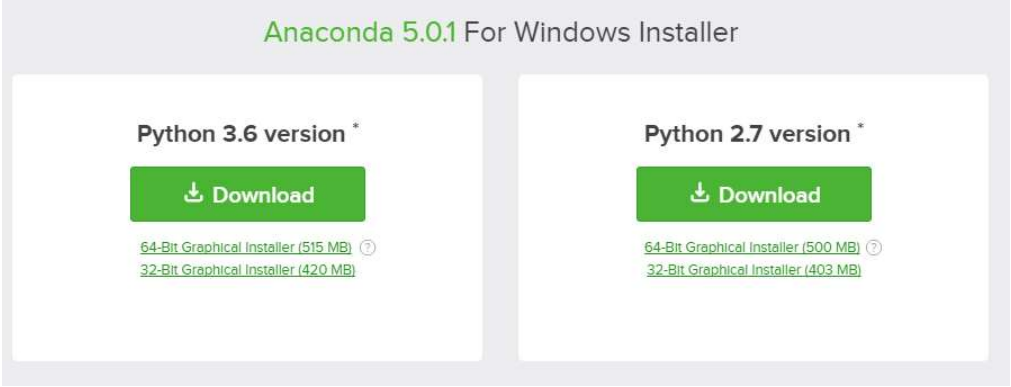
3. Download

Download the most recent Python 3 release. At the time of writing, the most recent release was the Python 3.6 Version. Python 2.7 is legacy Python. For problem solvers, select the Python 3.6 version. If you are unsure if your computer is running a 64-bit or 32-bit version of Windows, select 64-bit as 64- bit Windows is most common.

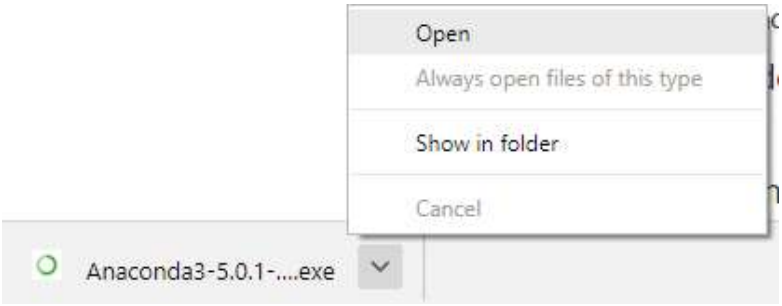
MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab



Once the download completes, open and run the ***.exe*** installer



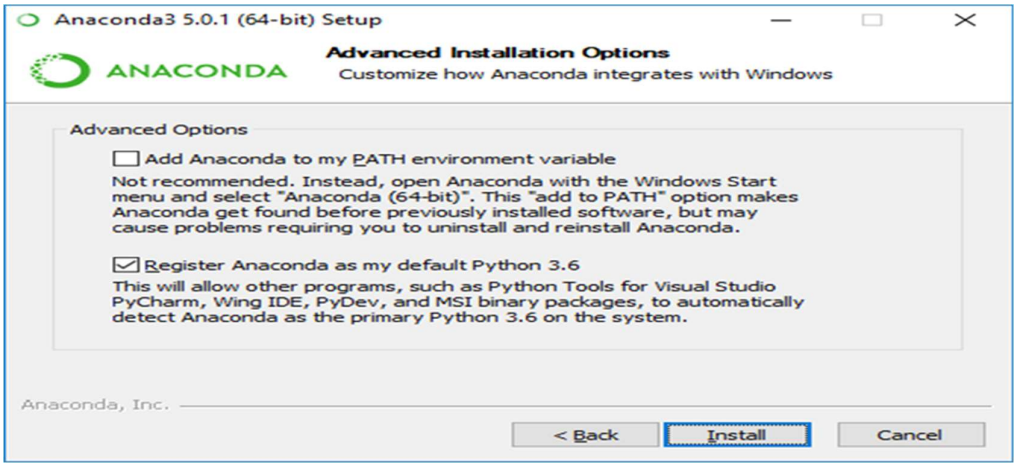
At the beginning of the install, you need to click Next to confirm the installation

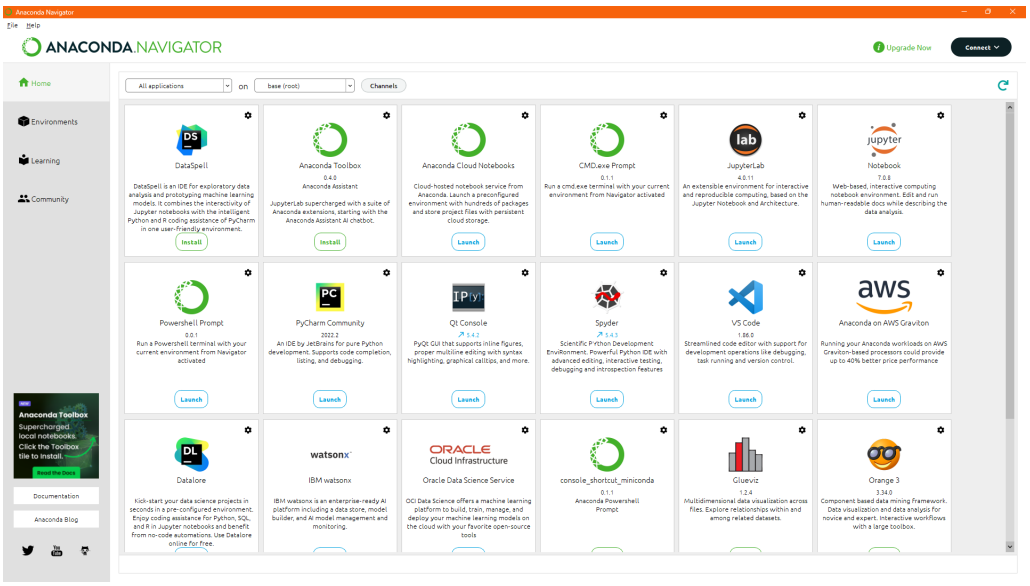
At the Advanced Installation Options screen, I recommend that you **do not check** "Add Anaconda to my PATH environment variable"

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015
(Maharashtra Act No. XXXIX of 2015)

Artificial Intelligence and Machine Learning Lab





MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Exercise for Practice	
Q1.	What are the key features of Python?
Ans:	Python is known for its simplicity, readability, and versatility. Its key features include dynamic typing, automatic memory management , and a rich standard library, making it an ideal choice for various applications.
Q2.	What is Python? List some popular applications of Python in the world of technology.
Ans	Python is a high-level programming language known for its simplicity and versatility. used in web development (Django, Flask), data science (Pandas, NumPy) artificial intelligence (TensorFlow, PyTorch), and automation (Selenium, BeautifulSoup),

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Q3	What are the benefits of using Python?
Ans	The benefits of using Python include its simplicity, readability, and vast ecosystem of libraries, making development faster and more efficient.
Q4.	What is the difference between Python Arrays and lists?
Ans	Lists are flexible and can hold elements of different data types, while arrays are homogeneous collections of elements of the same data type. Lists come with built-in methods for manipulation, whereas arrays require the NumPy library
Conclusion	Thus, we have successfully installed Anaconda on windows and Ubuntu; We also have described the libraries and the installation of the libraries.

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 06

Title: Download the any dataset from UCI or Data.org or from any other data repositories and perform the basic data pre-processing steps using python/R

Theory:

Step 1: Import the libraries

Step 2: Import the data-set

Step 3: Check out the missing values

Step 4: See the Categorical Values

Step 5: Splitting the data-set into Training and Test Set

Data cleaning:

The main aim of Data Cleaning is to identify and remove errors & duplicate data, in order to create a reliable dataset. This improves the quality of the training data for analytics and enables accurate decision-making.

Data cleansing is a time-consuming process and most data scientists spend an enormous amount of time in enhancing the quality of the data. However, there are various methods to identify and classify data for data cleansing.
There are mainly two distinct techniques, namely Qualitative and Quantitative techniques to classify data errors. Qualitative techniques involve rules, constraints, and patterns to identify errors.

On the other hand, Quantitative techniques employ statistical techniques to identify errors in the trained data.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Normalisation:

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

Here's the formula for normalization:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, Xmax and Xmin are the maximum and the minimum values of the feature respectively.

- When the value of X is the minimum value in the column, the numerator will be 0, and hence X' is 0
- On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator and thus the value of X' is 1
- If the value of X is between the minimum and the maximum value, then the value of X' is between 0 and 1

Standardisation:

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

Here's the formula for standardization:

$$X' = \frac{X - \mu}{\sigma}$$

μ is the mean of the feature values and σ is the standard deviation of the feature values. Note that in this case, the values are not restricted to a particular range.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

X_train,X_test,y_train,y_test=
train_test_split(X,y,test_size=0.2,random_state= 0)

Split arrays or matrices into random train and test subsets Quick utility that wraps input validation
and next(ShuffleSplit().split(X, y)) and application to input data into a single call for splitting (and optionally subsampling) data in oneliner.

Imputer(missing_values='NaN', strategy='mean', axis=0)

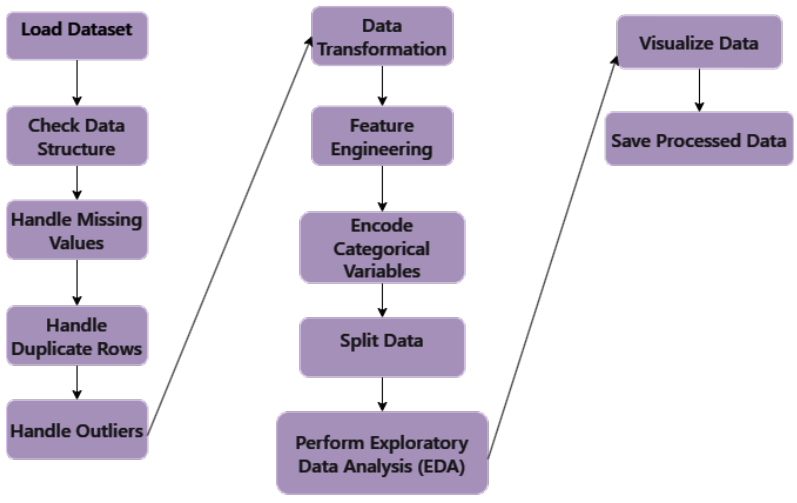
Imputation transformer for completing missing values.

pandas.read_csv()
Read a comma-separated values (csv) file into DataFrame.

Also supports optionally iterating or breaking of the file into chunks.

Flowchart:

(if any)



Pseudo code or Program

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

```
url = "https://archive.ics.uci.edu/
              ml/machine-learning-databases
              /iris/iris.data"

col_names = ['sepal_length', 'sepal_width',
             'petal_length', 'petal_width',
             'class']

iris_df = pd.read_csv(url, header=None,
          names=col_names)

print("Original Dataset:")
print(iris_df.head())

print("\nSummary Statistics:")
print(iris_df.describe())

print("\nMissing Values:")
print(iris_df.isnull().sum())

print("\nDuplicate Rows:")
print(iris_df.duplicated().sum())

print("\nData Types:")
print(iris_df.dtypes)

iris_df['class'] =
             iris_df['class'].astype('category')

print("\nUnique Classes:")
print(iris_df['class'].unique())

X = iris_df.drop('class', axis=1)
y = iris_df['class']
```

Outputs

```
Original Dataset:
   sepal_length  sepal_width  petal_length  petal_width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa

Summary Statistics:
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000

Missing Values:
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
class           0
dtype: int64

Duplicate Rows:
3

Data Types:
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
class            object
dtype: object

Unique Classes:
['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
Categories (3, object): ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

	(Students can attach extra pages if necessary)
Exercise for Practice	
Q1.	What is the difference between data processing and data mining?
Ans:	Data processing refers to the manipulation and transformation of raw data into meaningful information Data mining is the process of discovering patterns, trends, and insights from large datasets using statistical, machine learning techniques.
Q2.	What is the difference between supervised and unsupervised learning?
Ans	Supervised learning is a type of machine learning where the model learns from labeled data, meaning the training dataset contains input-output pairs Unsupervised learning is a type of machine learning where the model learns from unlabeled data, meaning the training dataset contains input features without corresponding output labels.
Q3	What is the difference between data cleaning and data transformation?
Ans	Data cleaning involves identifying and correcting errors, inconsistencies, and anomalies in the dataset to ensure its accuracy and reliability. Data transformation involves converting or modifying the original dataset to improve its quality, suitability, or compatibility for analysis or modeling purposes.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Q4.	What is data pre-processing, and why is it essential in data science?
Ans	Data preprocessing refers to the process of cleaning, transforming, and preparing raw data into a format that is suitable for analysis or modeling. It quality and suitability of the data directly impact the accuracy and reliability of the results.
Conclusion	Thus, we have successfully implemented pre-processing operations on a dataset

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 07

Title: Develop a Bayesian classifier.

Theory:

Bayes Theorem

Bayes' Theorem is a way of finding a probability when we know certain other probabilities.

The formula is:

$P(A|B) = P(A)\,P(B|A)\mathbf{P(B)}$

Which tells us:	how often A happens *given that B happens*, written **P(A|B)**,
When we know:	how often B happens *given that A happens*, written **P(B|A)**

and how likely A is on its own, written **P(A)**

and how likely B is on its own, written **P(B)**

Bayes Classifier with example

In machine learning, naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models.[1] But they could be coupled with Kernel density estimation and achieve higher accuracy levels.

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

	based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

Pseudo code or Program	`import pandas as pd`
	`from sklearn.datasets import load_breast_cancer`
	`from sklearn.model_selection`
	`import train_test_split, GridSearchCV`
	`from sklearn.naive_bayes import GaussianNB`
	`from sklearn.metrics import accuracy_score`
	`from sklearn.preprocessing import StandardScaler`
	
	`# Load breast cancer dataset from sklearn`
	`data = load_breast_cancer()`
	
	`# Convert to DataFrame`
	`df = pd.DataFrame(data.data, columns=data.feature_names)`
	`df['target'] = data.target`
	
	`# Display the first few rows of the dataset`
	`# to understand its structure`
	`print(df.head())`
	
	`# Separate features and target`
	`X = df.drop('target', axis=1) # Features`
	`y = df['target'] # Target`
	
	`# Standardize the features`
	`scaler = StandardScaler()`
	`X_scaled = scaler.fit_transform(X)`
	
	`# Split the dataset into training and test sets`
	`X_train, X_test, y_train, y_test =`
	`train_test_split(X_scaled, y,`
	`test_size=0.2,`
	`random_state=42)`
	
	`# Define the parameter grid`
	`param_grid = {`
	`'var_smoothing': [1e-9, 1e-8, 1e-7, 1e-6, 1e-5]`
	`}`

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

```
# Initialize the Gaussian Naive Bayes classifier
bayes = GaussianNB()

# Initialize GridSearchCV with
# 5-fold cross-validation
grid_search = GridSearchCV(estimator=bayes,
          param_grid=param_grid, cv=5)

# Perform hyperparameter tuning
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_params = grid_search.best_params_

# Train the model with the best hyperparameters
best_bayes = GaussianNB(**best_params)
best_bayes.fit(X_train, y_train)

# Predict on the test set
y_pred = best_bayes.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Test Accuracy:", accuracy)
```

Outputs

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst area  \
0                 0.07871  ...          17.33           184.60      2019.0
1                 0.05667  ...          23.41           158.80      1956.0
2                 0.05999  ...          25.53           152.50      1709.0
3                 0.09744  ...          26.50            98.87       567.7
4                 0.05883  ...          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
4            0.1374             0.2050           0.4000                0.1625

   worst symmetry  worst fractal dimension  target
0          0.4601                  0.11890       0
1          0.2750                  0.08902       0
2          0.3613                  0.08758       0
3          0.6638                  0.17300       0
4          0.2364                  0.07678       0

[5 rows x 31 columns]
Test Accuracy: 0.9649122807017544
```

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Exercise for Practice	
Q1.	How can Bayes classifier can be used for categorical features? What if some features are numerical?
Ans:	A Bayes classifier can handle categorical features by calculating the conditional probabilities of each category for a given class. For numerical features, it assumes a Gaussian distribution and uses the mean and standard deviation to estimate.
Q2.	What are advantages and disadvantages of Naive Bayes Algorithm?
Ans	It is simple, fast, and scalable, handles missing data well, and works well with high-dimensional sparse data and small datasets
Q3	How is Bayes' Theorem relevant to the field of artificial intelligence?
Ans	It is fundamental to AI as it provides a probabilistic framework for handling uncertainty, allowing AI models to update probabilities as new data becomes available, thereby improving prediction accuracy and used in NLP, decision making systems etc
Q4.	What are the Applications of Naive Bayes Classifier?
Ans	It is widely used in various applications such as credit scoring, medical data classification, and real-time predictions. It's highly effective in text classification tasks like spam filtering and sentiment analysis

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Q5	Discuss a situation where the naive Bayes' approach would not be appropriate to use.
Ans	The Naive Bayes' approach may not be appropriate when the assumption of feature independence does not hold, as it often doesn't in real-world applications It also struggles with the zero-frequency problem
Conclusion	Thus, we have successfully completed the implementation of Naïve Bayes Gaussian Classifier.

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Enrolment No: MITU21BTCS0489 **Division:** TY-CC-2-B

Roll No: 2213825 **Name:** Rudradev Arya

Experiment No.: 08

Title: Implement K means algorithm for multidimensional data for Cars or Wine dataset from UCI repository

Theory:

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labelled responses. The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data.

Common clustering algorithms include:

- Hierarchical clustering: builds a multilevel hierarchy of clusters by creating a cluster tree
- k-Means clustering: partitions data into k distinct clusters based on distance to the centroid of a cluster
- Gaussian mixture models: models clusters as a mixture of multivariate normal density components
- Self-organizing maps: uses neural networks that learn the topology and distribution of the data
- Hidden Markov models: uses observed data to recover the sequence of states

Unsupervised learning methods are used in bioinformatics for sequence analysis and genetic clustering; in data mining for sequence and pattern mining; in medical imaging for image segmentation; and in computer vision for object recognition.

Clustering and its types:

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

Let's understand this with an example. Suppose, you are the head of a rental store and wish to understand preferences of your costumers to scale up your business. Is it possible for you to look at details of each costumer and devise a unique business strategy for each one of them? Definitely not. But, what you can do is to cluster all of your costumers into say 10 groups based on their purchasing habits and use a separate strategy for costumers in each of these 10 groups. And this is what we call clustering.

A "clustering" is essentially a set of such clusters, usually containing all objects in the data set. Additionally, it may specify the relationship of the clusters to each other, for example, a hierarchy of clusters embedded in each other. Clustering's can be roughly distinguished as:

Hard clustering: each object belongs to a cluster or not

- Soft clustering (also: fuzzy clustering): each object belongs to each cluster to a certain degree (for example, a likelihood of belonging to the cluster)

Types:

Strict partitioning clustering: each object belongs to exactly one cluster

- Strict partitioning clustering with outliers: objects can also belong to no cluster, and are considered outliers
- Overlapping clustering (also: alternative clustering, multi-view clustering): objects may belong to more than one cluster; usually involving hard clusters
- Hierarchical clustering: objects that belong to a child cluster also belong to the parent cluster
- Subspace clustering: while an overlapping clustering, within a uniquely defined subspace, clusters are not expected to overlap

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

K means algorithm

Kmeans algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

The way k means algorithm works is as follows:

1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.

- Compute the sum of the squared distance between data points and all centroids.
- Assign each data point to the closest cluster (centroid).
- Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

The approach k-means follows to solve the problem is called Expectation- Maximization.

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

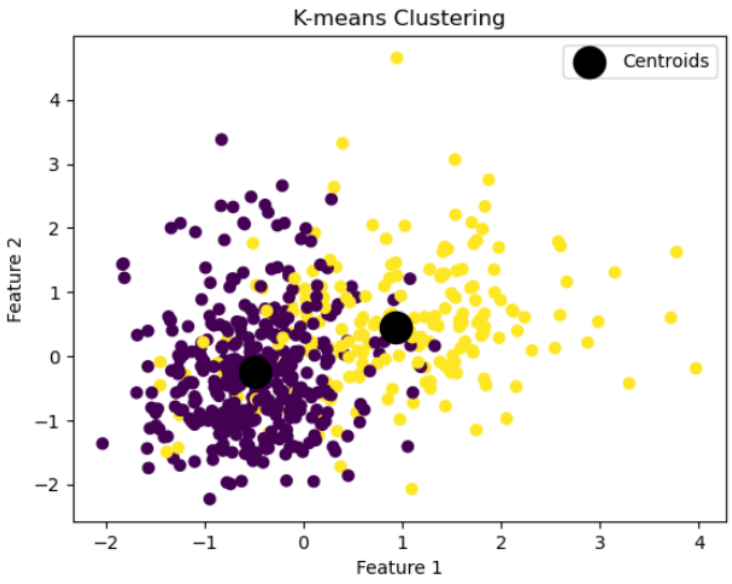
Pseudo code or Program

```
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.datasets import load_breast_cancer
# Load Breast Cancer dataset
cancer_data = load_breast_cancer()
df = pd.DataFrame(data=cancer_data.data,
columns=cancer_data.feature_names)
# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df)
# Perform K-means clustering
kmeans = KMeans(n_clusters=2, max_iter=400,
            verbose=True, tol=0.2)
pred = kmeans.fit_predict(X_scaled)
# Calculate silhouette score
silhouette_avg = silhouette_score(X_scaled, pred)
print(f"Silhouette Score: {silhouette_avg}")
# Plotting the clusters
plt.scatter(X_scaled[:, 0], X_scaled[:, 1],
           c=pred, cmap='viridis')
plt.scatter(kmeans.cluster_centers_[:, 0],
            kmeans.cluster_centers_[:, 1],
s=300, c='black', label='Centroids')
plt.title('K-means Clustering')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.legend()
plt.show()Type your text
```

Outputs

(Students can attach extra pages if necessary)

```
Initialization complete
Iteration 0, inertia 22538.974789999458.
Iteration 1, inertia 11872.241180317718.
Iteration 2, inertia 11612.43523650218.
Converged at iteration 2: center shift 0.039205508005132 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 18020.702340193602.
Iteration 1, inertia 13776.429269118165.
Iteration 2, inertia 11897.766175578681.
Iteration 3, inertia 11611.771642170908.
Converged at iteration 3: center shift 0.04545228602696646 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 19055.9348477484.
Iteration 1, inertia 12568.210098894124.
Iteration 2, inertia 11910.585509832781.
Iteration 3, inertia 11666.88290834656.
Converged at iteration 3: center shift 0.16512240137922354 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 15647.500365072587.
Iteration 1, inertia 12030.102282729038.
Iteration 2, inertia 11704.285720363203.
Iteration 3, inertia 11617.405462937697.
Converged at iteration 3: center shift 0.03560345694391448 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 19839.128609678075.
Iteration 1, inertia 11643.344045303764.
Converged at iteration 1: center shift 0.10822450210183912 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 17907.893484556218.
Iteration 1, inertia 11716.2997035144.
Iteration 2, inertia 11622.274188699981.
Converged at iteration 2: center shift 0.04962584177811263 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 21288.66307815267.
Iteration 1, inertia 12141.084506025307.
Iteration 2, inertia 11626.155895427544.
Converged at iteration 2: center shift 0.09858568139014133 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 21887.06303014562.
Iteration 1, inertia 13474.1344376058.
Iteration 2, inertia 12258.961344186948.
Iteration 3, inertia 11692.136565510984.
Iteration 4, inertia 11619.460503183815.
Converged at iteration 4: center shift 0.043163149155173906 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 17655.178713896483.
Iteration 1, inertia 12136.374365138498.
Iteration 2, inertia 11683.773545252929.
Iteration 3, inertia 11600.609300457287.
Converged at iteration 3: center shift 0.02152823278550034 within tolerance 0.20000000000000007.
Initialization complete
Iteration 0, inertia 18865.01567002759.
Iteration 1, inertia 15178.011183261746.
Iteration 2, inertia 14332.020931115414.
Iteration 3, inertia 12445.843440340237.
Iteration 4, inertia 11664.110527019835.
Iteration 5, inertia 11600.110245443406.
Converged at iteration 5: center shift 0.018012005612583763 within tolerance 0.20000000000000007.
Silhouette Score: 0.3433822406907781
```



MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Exercise for Practice

Q1.	What are the advantages and disadvantages of K-means Algorithm?
Ans:	The K-means Algorithm is simple, scalable, and guarantees convergence, making it suitable for large datasets However, it requires pre-specification of the number of clusters is sensitive to initial centroid selection and outliers, and may struggle with clusters of varying sizes and densities
Q2.	What are some stopping criteria of K-means clustering?
Ans	1) The centroids of newly formed clusters do not changeType your text 2) The data points remain in the same cluster 3) The maximum number of iterations is reached
Q3	What is the difference between K-means and K-nearest neighbours?
Ans	K-means is an unsupervised learning algorithm used for clustering, where 'K' refers to the number of clusters It iteratively calculates the distances between data points and cluster centroids, aiming to minimize the overall variance within each cluster. On the other hand, K-nearest neighbours (KNN) is a supervised learning algorithm used for classification and regression, where 'K' is the number of nearest neighbors . It directly calculates the distances between data points
Q4.	Explain some cases where K-means clustering fails to give good reasons.
Ans	K-means clustering can fail in cases where clusters are of varying sizes and densities, as it struggles to accurately assign data points to the correct clusters . It also performs poorly with non-convex shapes and is sensitive to outliers. Furthermore, it requires the number of clusters (k) to be specified in advance, which may not always be known

MIT Art Design and Technology University's
MIT School of Computing, Pune
Department of Computer Science and Engineering
BTech Third Year **A.Y.2023-24**

MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015 (Maharashtra Act No. XXXIX of 2015))

Artificial Intelligence and Machine Learning Lab

Conclusion	Thus, we have successfully completed the implementation of K-means algorithm

Date of Submission:	
Marks out of 10	**Name and Sign of Subject Teacher**
Remark:	

Courtesy:

https://geekflare.com/tic-tac-toe-python-code/

https://favtutor.com/blogs/breadth-first-search-python

https://www.mygreatlearning.com/blog/a-search-algorithm-in-artificial-intelligence/

https://www.anaconda.com/download

https://archive.ics.uci.edu/datasets

https://data.gov/

https://www.analyticsvidhya.com/blog/2021/01/a-guide-to-the-naive-bayes-algorithm/