Parle Tilak Vidyalaya Associations

SATHAYE COLLEGE (Autonomous)

Vile-Parle (East), Mumbai – 400 057.

Practical Journal

Applied Artificial Intelligence

Submitted by

Mazhar Iqbal Solkar
Seat No : 27

M.Sc. [I.T.]-Information Technology Part II
2022 – 2023

Parle Tilak Vidyalaya Association's

SATHAYE COLLEGE (Autonomous)

Vile-Parle (East), Mumbai – 400 057.

CERTIFICATE

This is to certify that **Mazhar Iqbal Solkar**

Seat No **27** has successfully completed all the practicals in the subject of **Applied Artificial Intelligence** for M.Sc.I.T. Part-II SEM – III as prescribed by University of Mumbai for the year 2022-2023.

Coordinator M.Sc. [I.T.]	Professor in Charge	External Examiner
Date:	Date	Date

INDEX

Sr.No.	Date	Practical Title	Sign
1		Design an Expert sytem using AIML	
2		Design a bot using AIML	
3		Implement Bayes Theorem using python	
4		Implement Conditional Probability and Joint Probability using python	
5		A program to implement Rule Based System	
6		Design a Fuzzy based application using python	
7		Write an application to simulate supervised and un-supervised learning model	
8		Write an application to implement clustering algorithm	
9		Write a program to implement BFS algorithm	
10		Write a program to implement DFS algorithm	
11		Write an program to implement support vector machine	
12		Write a program to implement parser in natural language processing	

Practical : 1

Aim : Design an Expert system using AIML.

Description :

What is an Expert System?

An expert system is a computer program that is designed to solve complex problems and to provide decision-making ability like a human expert. It performs this by extracting knowledge from its knowledge base using the reasoning and inference rules according to the user queries.

Code :

hi.aiml M Practical1_AI.py M std-startup.xm

Practical1_AI.py > ...

```python
import aiml
import time
time.clock=time.time
kernel = aiml.Kernel()
kernel.learn("std-startup.xml")
kernel.respond("LOAD AIML B")
while True:
    input_text = input(">Human: ")
    response = kernel.respond(input_text)
    print(">Bot: "+response)
```

hi.aiml M Practical1_AI.py M std-startup.xml M

std-startup.xml

```xml
<aiml version= "1.0.1" encoding="UTF-8">
    <category>
        <pattern>LOAD AIML B</pattern>
        <template>
            <learn>hi.aiml</learn>
        </template>
    </category>
</aiml>
```

hi.aiml M

hi.aiml

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>

<aiml version="1.0.1">



<category>
<pattern>HI</pattern>
<template>
<random>
<li>Hello there!!</li>
<li>Hey</li>
</random>
</template>
</category>

<category>
<pattern>SUNDAY</pattern>
<template>the day of the week before Monday and following Saturday</template>
</category>
<category>
<pattern>MONDAY</pattern>
<template>the day of the week before Tuesday and following Sunday</template>
</category>
<category>
<pattern>TUESDAY</pattern>
<template>the day of the week before Wednesday and following Monday</template>
</category>
<category>
<pattern>WEDNESDAY</pattern>
<template>the day of the week before Thursday and following Tuesday</template>
</category>
<category>
<pattern>THURSDAY</pattern>
<template>the day of the week before Friday and following Wednesday</template>
</category>
<category>
<pattern>FRIDAY</pattern>
<template>the day of the week before Saturday and following Thursday</template>
</category>
<category>
<pattern>SATURDAY</pattern>
<template>the day of the week before Sunday and following Friday</template>
</category>



</aiml>
```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence\practical_01> python -u "c:\MSC-IT\Applied
Loading std-startup.xml...done (0.10 seconds)
Loading hi.aiml...done (0.00 seconds)
>Human: hi
>Bot: Hello there!!
>Human: monday
>Bot: the day of the week before Tuesday and following Sunday
>Human: tuesday
>Bot: the day of the week before Wednesday and following Monday
>Human: wednesday
>Bot: the day of the week before Thursday and following Tuesday
>Human: thursday
>Bot: the day of the week before Friday and following Wednesday
>Human: friday
>Bot: the day of the week before Saturday and following Thursday
>Human: saturday
>Bot: the day of the week before Sunday and following Friday
>Human: sunday
>Bot: the day of the week before Monday and following Saturday
>Human:
```

Practical : 2

Aim : Design a bot using AIML.

Description :

What is AIML?

AIML stands for Artificial Intelligence Modelling Language. AIML is an XML based markup language meant to create artificial intelligent applications. AIML makes itpossible to create human interfaces while keeping the implementation simple to program, easy to understand and highly maintainable. This tutorial will teach you the basics of AIML. All the basic components of AIML with suitable examples havebeen discussed in this tutorial.

AIML Tags/Description

- <aiml> − defines the beginning and end of a AIML document.
- <category> − defines the unit of knowledge in bot's knowledge base.
- <pattern> − defines the pattern to match what a user may input to an bot.
- <template> − defines the response of a bot to user's input.

Code :

Practical1_AI.py M

Practical1_AI.py > ...

```python
import aiml
import time
time.clock=time.time
kernel = aiml.Kernel()
kernel.learn("std-startup.xml")
kernel.respond("LOAD AIML B")
while True:
    input_text = input(">Human: ")
    response = kernel.respond(input_text)
    print(">Bot: "+response)
```

std-startup.xml M

std-startup.xml

```xml
<aiml version= "1.0.1" encoding="UTF-8">
      <category>
           <pattern>LOAD AIML B</pattern>
           <template>
                <learn>hi.aiml</learn>
            </template>
      </category>
</aiml>
```

hi.aiml M × Practical1_AI.py M

hi.aiml

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>

<aiml version="1.0.1">


<category>
<pattern>HI</pattern>
<template>
<random>
<li>Hello there!!</li>
<li>Hey</li>
</random>
</template>
</category>

<category>
<pattern>WHAT IS YOUR NAME</pattern>
<template>My name is Siri</template>
</category>

<category>
<pattern>WHAT IS AIML</pattern>
<template>AIML is a language</template>
</category>

<category>
<pattern>WHAT IS *</pattern>
<template><set name ="username"> <star /></set></template>
</category>

<category>
<pattern>WHAT ABOUT MOVIES</pattern>
<template>Do you like comedy movies</template>
</category>

<category>
<pattern>YES</pattern>
<that>Do you like comedy movies</that>
<template>Nice, I like comedy movies too.</template>
</category>

<category>
<pattern>NO</pattern>
<that>Do you like comedy movies</that>
<template>Ok! But I like comedy movies.</template>
</category>

</aiml>

```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence\practical_01> python -u "c:\MSC-IT\Applied Artificial Intel
Loading std-startup.xml...done (0.08 seconds)
Loading hi.aiml...done (0.00 seconds)
>Human: hi
>Bot: Hello there!!
>Human: what is your name
>Bot: My name is Siri
>Human: what is aiml
>Bot: AIML is a language
```

Practical : 3

Aim : Implement Bayes Theorem using Python.

Description :

Bayes' Theorem provides a way that we can calculate the probability of a piece ofdata belonging to a given class, given our prior knowledge. Bayes' Theorem is stated as:

P(class|data) = (P(data|class) * P(class)) / P(data)

Where P(class|data) is the probability of class given the provided data.
Naive Bayes is a classification algorithm for binary (two-class) and multiclassclassification problems. It is called Naive Bayes or idiot Bayes because the calculations of the probabilities for each class are simplified to make their calculations tractable.

Code :

aaiPractical_3_BayesTheorem.py - C:/Users/mazhar/AppData/Local/Programs/Python/Python310/aaiPractical_3_BayesTheorem.py (3.10.8)

File Edit Format Run Options Window Help

```
def are_drug_users(n, users_of_drug):
    m = 1-n
    non_users_of_drug = 1 - users_of_drug
    return (n * users_of_drug)/(n * users_of_drug + m * non_users_of_drug)

print(are_drug_users(0.99,0.005))
```

Output :

IDLE Shell 3.10.8

File Edit Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/mazhar/AppData/Local/Programs/Python/Python310/aaiPractical_
3_BayesTheorem.py
0.33221476510067094
```

Practical : 4

Aim : Implement Conditional Probability and joint probability using Python.

Description :

What is Conditional Probability?

The probability of one event given the occurrence of another event is called the conditional probability. The conditional probability of one to one or more randomvariables is referred to as the conditional probability distribution.

For example, the conditional probability of event A given event B is writtenformally as:

- **P(A given B)**

The "given" is denoted using the pipe "|" operator; for example:

- **P(A | B)**

The conditional probability for events A given event B is calculated as follows:

- **P(A given B) = P(A and B) / P(B)**

Code :

practical_2B.py U ×

practical_2B.py > ...

```python
pass_stats = 0.15
pass_coding_with_stats = 0.60   #pass_coding_With_Stats
pass_coding_without_stats = 0.40
p_both = pass_stats * pass_coding_with_stats
print("joint probability is = ",p_both)
p_coding = p_both + ((1-pass_stats)*pass_coding_without_stats)

stats_given_coding = p_both /p_coding
print("conditional probability is = ",stats_given_coding)
```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence\Practical_04> python -u "c
joint probability is =  0.09
conditional probability is =  0.2093023255813953
PS C:\MSC-IT\Applied Artificial Intelligence\Practical_04>
```

Description :

What is Joint Probability?

The probability of two (or more) events is called the joint probability. The joint probability of two or more random variables is referred to as the joint probability distribution.The joint probability for events A and B is calculated as the probabilityof event A given event B multiplied by the probability of event B.

This can be stated formally as follows:

P(A and B) = P(A given B) * P(B)

The calculation of the joint probability is sometimes called the fundamental rule of probability or the "product rule" of probability or the "chain rule" of probability
P(A and B) = P(A given B) * P(B) = P(B given A) * P(A)

Code :

practical_2B.py U ×

practical_2B.py > ...

```python
pass_stats = 0.15
pass_coding_with_stats = 0.60   #pass_coding_With_Stats
pass_coding_without_stats = 0.40
p_both = pass_stats * pass_coding_with_stats
print("joint probability is = ",p_both)
p_coding = p_both + ((1-pass_stats)*pass_coding_without_stats)

stats_given_coding = p_both /p_coding
print("conditional probability is = ",stats_given_coding)

```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence\Practical_04> python -u "c
joint probability is =  0.09
conditional probability is =  0.2093023255813953
PS C:\MSC-IT\Applied Artificial Intelligence\Practical_04>
```

Practical : 5

Aim : A program to implement Rule Based System.

Description :

What is Rule Based System?

A rule-based system is a system that applies human-made rules to store, sort andmanipulate data. In doing so, it mimics human intelligence.

To work, rule-based systems require a set of facts or source of data, and a set ofrules for manipulating that data. These rules are sometimes referred to as 'If statements' as they tend to follow the line of 'IF X happens THEN do Y'.

Automation software like Think Automation is a good example. It automatesprocesses by breaking them down into steps.

- First comes the data or new business event
- Then comes the analysis: the part where the system conditonally processes the data against its rules
- Then comes any subsequent automated follow-up actions

Code :

aaiprologpract.pl

C: > Users > mazhar > Desktop > aaiprologpract.pl

```
go:-
hypothesis(Disease),
write('I believe you have: '),
write(Disease),
nl,
undo.

hypothesis(cold) :- cold.
hypothesis(flu) :- flu.

cold :-
verify(headache),
verify(runny_nose),
verify(sneezing),
verify(sore_throat),
nl.

flu :-
verify(fever),
verify(headache),
verify(chills),
verify(body_ache),
nl.

ask(Question) :-
write('Does the patient have following symptom: '),
write(Question),
write('?'),
read(Response),
nl,
( (Response == yes ; Response == y)
->
assert(yes(Question));
assert(no(Question)), fail).

:- dynamic yes/1,no/1.
verify(S) :-
 (yes(S)
  ->
   true ;
    (no(S)
     ->
      fail;
       ask(S))).

undo :- retract(yes(_)),fail.
undo :- retract(no(_)),fail.
undo.
```

Output :

SWI-Prolog -- c:/Users/mazhar/Desktop/aaiprologpract.pl

File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.3)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- go.
Does the patient have following symptom: headache? y.

Does the patient have following symptom: runny_nose?|: y.

Does the patient have following symptom: sneezing?|: y.

Does the patient have following symptom: sore_throat?|: y.


I believe you have: cold
true .
```

Practical : 6

Aim : Design a Fuzzy based application using Python / R

Description :

What is Fuzzy based application?

Fuzzy sets were introduced by Lotfi Zadeh (1921–2017) in 1965.

Unlike crisp sets, a fuzzy set allows partial belonging to a set, that is defined by adegree of membership, denoted by μ, that can take any value from 0 (element does not belong at all in the set) to 1 (element belongs fully to the set).

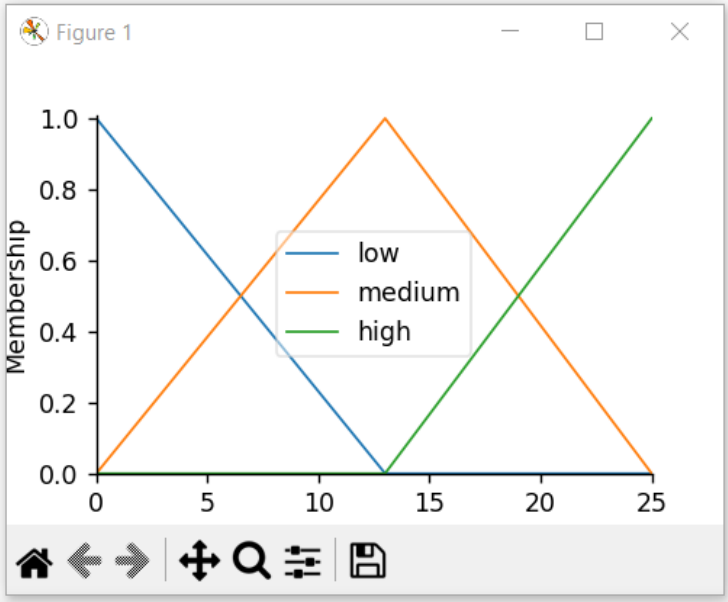
It is evident that if we remove all the values of belonging except from 0 and 1, thefuzzy set will collapse to a crisp set that was described in the previous section.

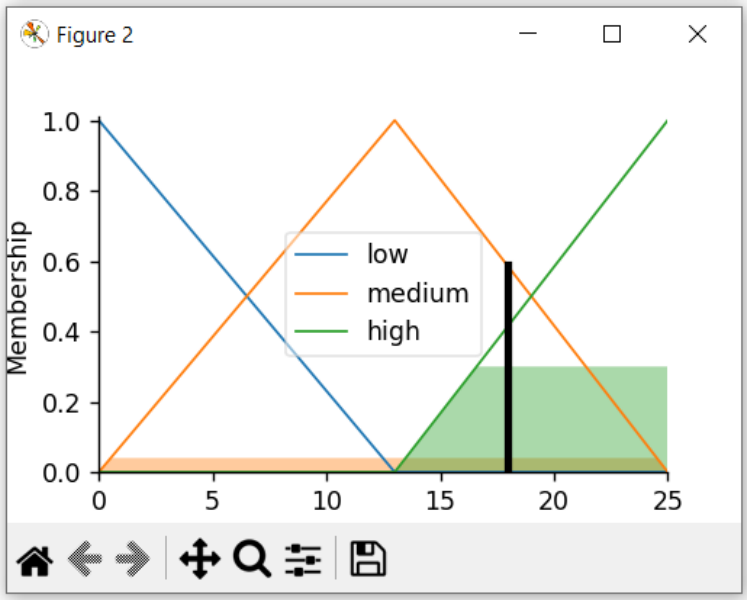
Code :

```
import numpy as np
import skfuzzy as fuzz     #pip install scikit-fuzzy
from skfuzzy import control as ctrl
quality = ctrl.Antecedent(np.arange(0,11,1),'quality')
service = ctrl.Antecedent(np.arange(0,11,1),'service')
tip = ctrl.Consequent(np.arange(0,26,1),'tip')
quality.automf(3)
service.automf(3)
tip['low']=fuzz.trimf(tip.universe,[0,0,13])
tip['medium']=fuzz.trimf(tip.universe,[0,13,25])
tip['high']=fuzz.trimf(tip.universe,[13,25,25])
tip.view()
rule1=ctrl.Rule(quality['poor']& service['poor'],tip['low'])
rule2=ctrl.Rule(quality['average']& service['average'],tip['medium'])
rule3=ctrl.Rule(quality['good']& service['good'],tip['high'])
tipping_ctrl=ctrl.ControlSystem([rule1,rule2,rule3])
tipping=ctrl.ControlSystemSimulation(tipping_ctrl)
tipping.input['quality']=6.5
tipping.input['service']=9.8
tipping.compute()
print(tipping.output['tip'])
tip.view(sim=tipping)
```

Output :

```
Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug  1 2022, 21:53:49) [MSC v.1932 64 bit (AMD
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:\MSC-IT\Applied Artificial Intelligence\Practical_04\Fuzzification.py
17.93427234232227
```





Practical : 7

Aim : Write an application to simulate supervised and un-supervised learning model.

Description :

What is supervised learning?

Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically, supervised learning is a learning in which we teach or train themachine using data which is well labelled that means some data is already taggedwith the correct answer.

Supervised learning classified into two categories of algorithms:

- Classification: A classification problem is when the output variable is a category, such as “Red” or “blue” or “disease” and “no disease”.
- Regression: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Supervised learning deals with or learns with “labelled” data. Which implies that some data is already tagged with the correct answer.Types: -

- **Regression**
- **Logistic Regression**
- **Classification**
- **Naive Bayes Classifiers**
- **K-NN (k nearest neighbours)**
- **Decision Trees**
- **Support Vector Machine**

Code:

knn.py - C:/Users/mazhar/Desktop/knn.py (3.10.8)

File Edit Format Run Options Window Help

```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
import numpy as np
import matplotlib.pyplot as plt

irisData = load_iris()

# Create feature and target arrays
X = irisData.data
y = irisData.target

# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, rando

neighbors = np.arange(1, 9)
train_accuracy = np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))

# Loop over K values
for i, k in enumerate(neighbors):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

# Compute training and test data accuracy
train_accuracy[i] = knn.score(X_train, y_train)
test_accuracy[i] = knn.score(X_test, y_test)

# Generate plot
plt.plot(neighbors, test_accuracy, label = 'Testing dataset Accuracy')
plt.plot(neighbors, train_accuracy, label = 'Training dataset Accuracy')

plt.legend()
plt.xlabel('n_neighbors')
plt.ylabel('Accuracy')
plt.show()
```

Output :

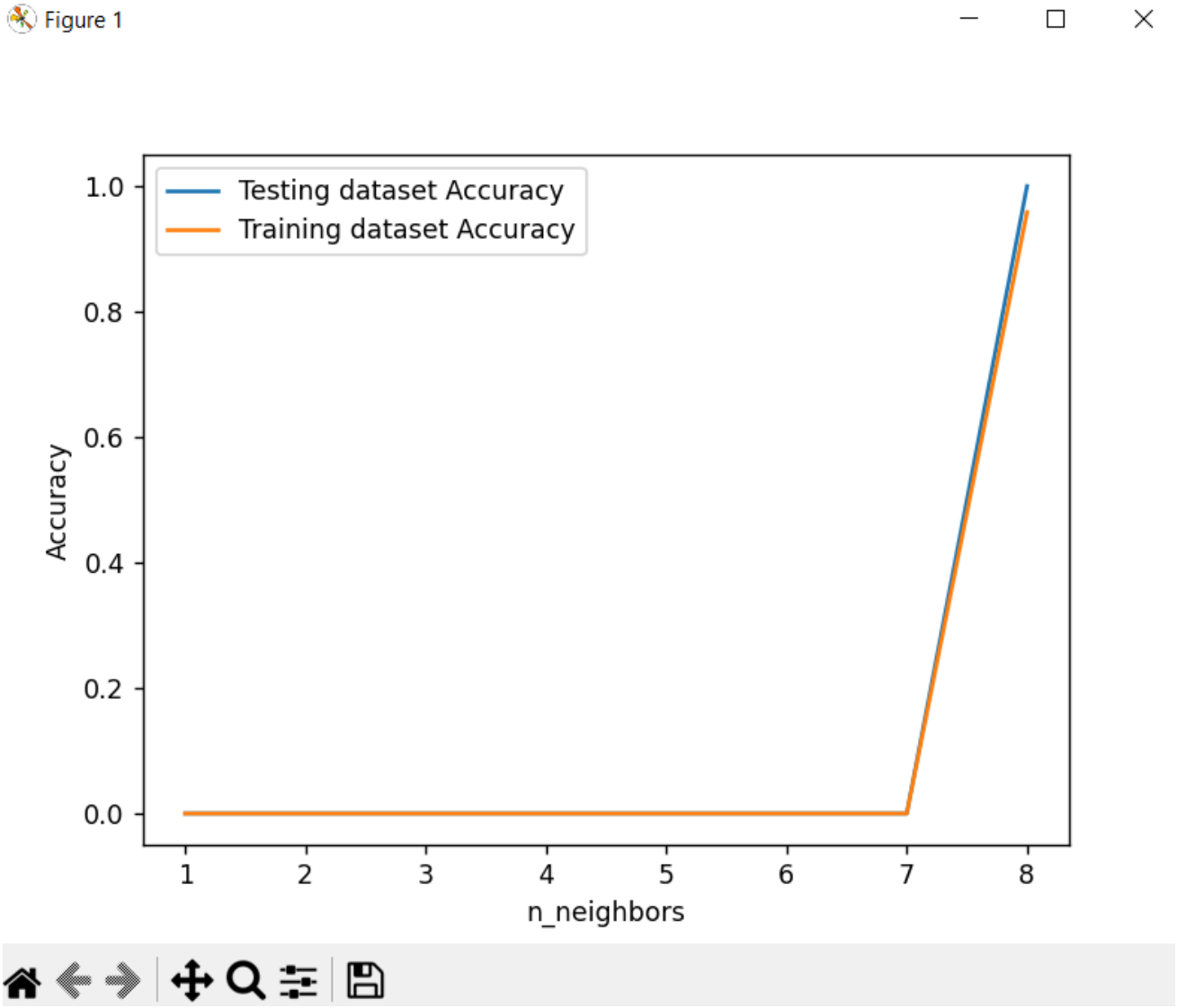


Figure 1
Testing dataset Accuracy
Training dataset Accuracy
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
7
8
n_neighbors

Description :

What is Unsupervised Learning?

Unsupervised learning is the training of machine using information that is neitherclassified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unsupervised learning classified into two categories of algorithms:

- Clustering: A clustering problem is where you want to discover the inherentgroupings in the data, such as grouping customers by purchasing behaviour.
- Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy Xalso tend to buy Y.

Types of Unsupervised Learning: -

Clustering

- Exclusive (partitioning)
- Agglomerative
- Overlapping
- Probabilistic

Clustering Types: -

- Hierarchical clustering
- K-means clustering
- Principal Component Analysis
- Singular Value Decomposition
- Independent Component Analysis

Code:

kmeans.py - C:/Users/mazhar/Desktop/kmeans.py (3.10.8)

File Edit Format Run Options Window Help

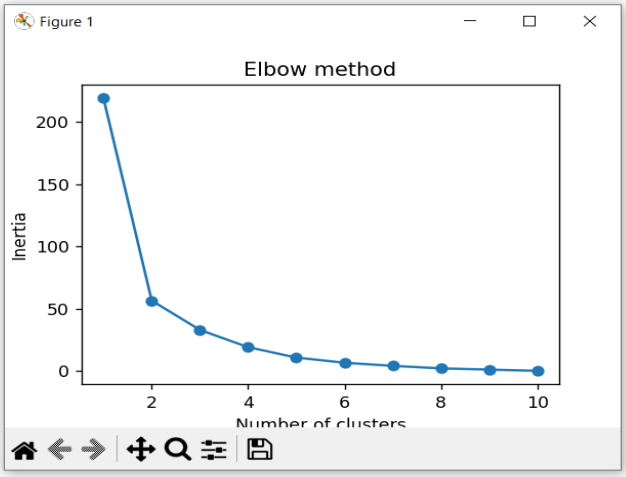
```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
data = list(zip(x, y))
print(data)
inertias = []

for i in range(1,11):
    kmeans = KMeans(n_clusters=i)
    kmeans.fit(data)
    inertias.append(kmeans.inertia_)

plt.plot(range(1,11), inertias, marker='o')
plt.title('Elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.show()
kmeans = KMeans(n_clusters=2)
kmeans.fit(data)

plt.scatter(x, y, c=kmeans.labels_)
plt.show()
```

Output :

IDLE Shell 3.10.8

File Edit Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:/Users/mazhar/Desktop/kmeans.py ==================
[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (6, 22), (10,
 21), (12, 21)]
```



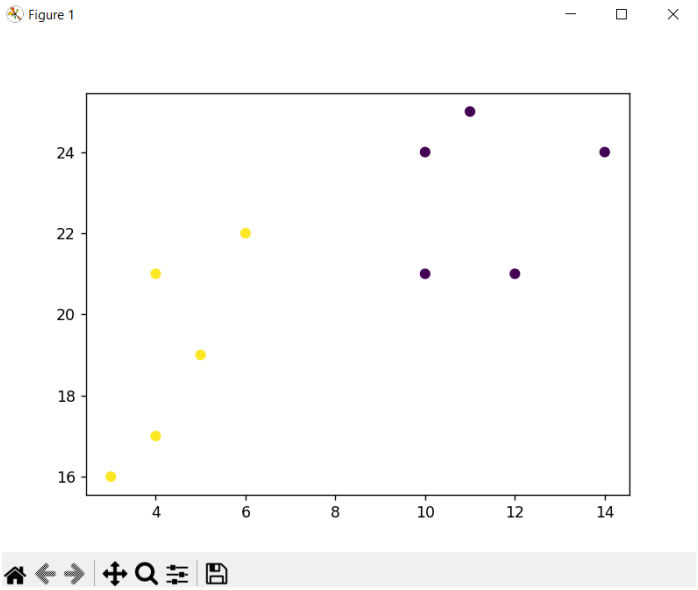


Figure 1
24
22
20
18
16
4
6
8
10
12
14

Practical : 8

Aim : Write an application to implement Clustering algorithm.

Description :

What is Clustering?

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 datapoints in the same group than those in other groups.

Code :

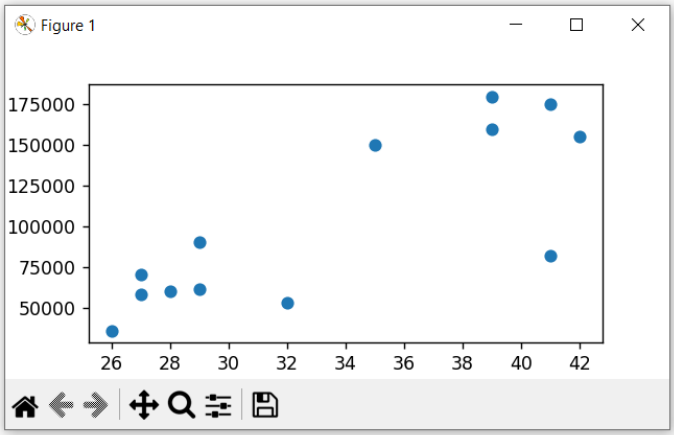
Clustering.py - C:\MSC-IT\Applied Artificial Intelligence\clustering\Clustering.py (3.10.8)

File Edit Format Run Options Window Help

```
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
df=pd.read_csv(r"C:/MSC-IT/Applied Artificial Intelligence/clustering/clusterDemo.csv")
print(df)
print("#######################################################")
plt.scatter(df['Age'],df['Income[$]'])
plt.show()
km = KMeans(n_clusters=2)
print(km)
print("#######################################################")
y_predicted=km.fit_predict(df[['Age','Income[$]']])
print(y_predicted)
print("#######################################################")
df['cluster']=y_predicted
print(df.head)
print("#######################################################")
df1=df[df.cluster==0]
df2=df[df.cluster==1]
plt.scatter(df1.Age,df1['Income[$]'],color='green')
plt.scatter(df2.Age,df2['Income[$]'],color='red')
plt.xlabel('Age')
plt.xlabel('income[$]')
plt.show()
```

Output :

```
*IDLE Shell 3.10.8*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\MSC-IT\Applied Artificial Intelligence\clustering\Clustering.py
        Name  Age  Income[$]
0        Rob   27      70000
1       Mani   29      90000
2      Mohan   29      61000
3       Kory   28      60000
4   Angelina   35     150000
5      Stark   42     155000
6   Abhishek   39     160000
7       Mary   41     175000
8        Sid   26      35800
9      Abdul   32      53000
10    Dipika   41      82000
11       Sam   27      58000
12     Hanry   39     180000
#######################################################
```




```
KMeans(n_clusters=2)
#######################################################
[1 1 1 1 0 0 0 0 1 1 1 1 0]
#######################################################
<bound method NDFrame.head of         Name  Age  Income[$]  cluster
0        Rob   27      70000        1
1       Mani   29      90000        1
2      Mohan   29      61000        1
3       Kory   28      60000        1
4   Angelina   35     150000        0
5      Stark   42     155000        0
6   Abhishek   39     160000        0
7       Mary   41     175000        0
8        Sid   26      35800        1
9      Abdul   32      53000        1
10    Dipika   41      82000        1
11       Sam   27      58000        1
12     Hanry   39     180000        0>
#######################################################
```

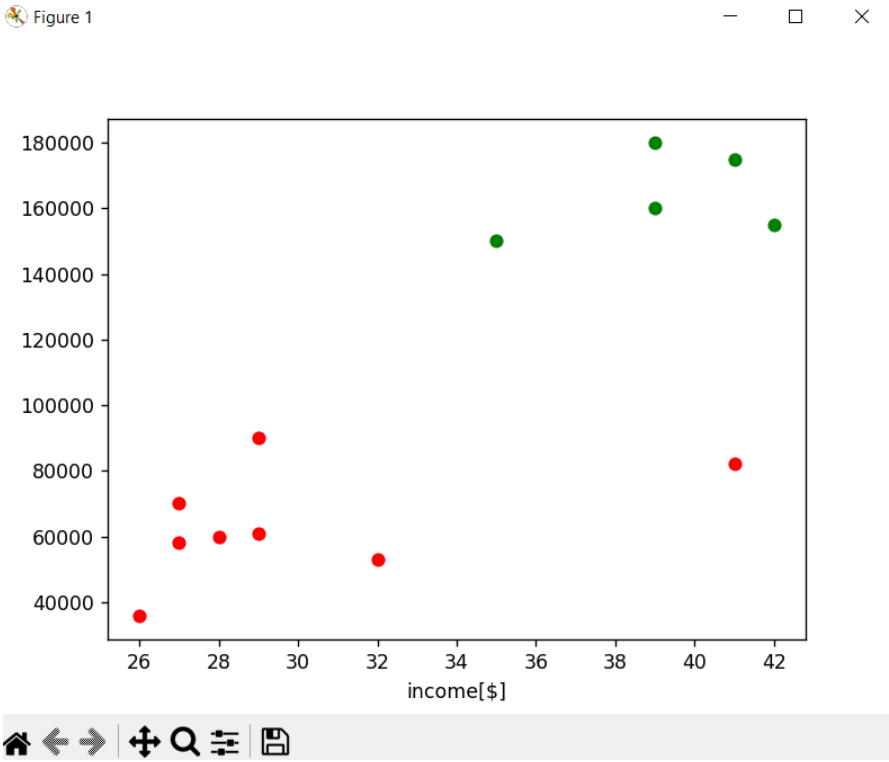


Figure 1
180000
160000
140000
120000
100000
80000
60000
40000
26
28
30
32
34
36
38
40
42
income[$]

Practical : 9

Aim : Write an Program to implement BFS algorithm.

Description :

What is Breadth-First Search?

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. As breadth-first search is the process of traversing each node of the graph, a standard BFS algorithm traverses each vertex of the graph into two parts:

1) Visited

2) Not Visited. So, the purpose of the algorithm is to visit all the vertex while avoiding cycles.

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.

Code :

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

visited = [] # List for visited nodes.
queue = []     #Initialize a queue

def bfs(visited, graph, node): #function for BFS
  visited.append(node)
  queue.append(node)

  while queue:          # Creating loop to visit each node
    m = queue.pop(0)
    print (m, end = " ")

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

# Driver Code
print("Following is the Breadth-First Search")
bfs(visited, graph, '5')    # function calling
```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence> & C:/Users/mazhar
elligence/bfs/bfs.py"
Following is the Breadth-First Search
5 3 7 2 4 8
PS C:\MSC-IT\Applied Artificial Intelligence>
```

Practical : 10

Aim : Write an Program to implement DFS algorithm.

Description :

What is Depth-First Search ?

The Depth-First Search is a recursive algorithm that uses the concept of backtracking. It involves thorough searches of all the nodes by going ahead if potential, else by backtracking. Here, the word backtrack means once you are moving forward and there are not any more nodes along the present path, youprogress backward on an equivalent path to seek out nodes to traverse.

Algorithm:

- Create a recursive function that takes the index of the node and a visitedarray.
- Mark the current node as visited and print the node.
- Traverse all the adjacent and unmarked nodes and call the recursive function with the index of the adjacent node.

Code :

```
# Using a Python dictionary to act as an adjacency list
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : ['8'],
  '8' : []
}

visited = set() # Set to keep track of visited nodes of graph.

def dfs(visited, graph, node):  #function for dfs
    if node not in visited:
        print (node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

# Driver Code
print("Following is the Depth-First Search")
dfs(visited, graph, '5')
```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence> & C:/Users/mazha
elligence/dfs/dfs.py"
Following is the Depth-First Search
5
3
2
4
8
7
PS C:\MSC-IT\Applied Artificial Intelligence>
```

Practical : 11

Aim : Write an Program to implement support vector machine

Description :

Support Vector Machine is a discriminative classifier that is formally designed by a separative hyperplane. It is a representation of examples as points in space that are mapped so that the points of different categories are separated by a gap as wide as possible. In addition to this, an SVM can also perform non-linear classification. Let us take a look at how the Support Vector Machine work.

The main objective of a support vector machine is to segregate the given data in the best possible way. When the segregation is done, the distance between the nearest points is known as the margin. The approach is to select a hyperplane with the maximum possible margin between the support vectors in the given data-sets.

An SVM kernel basically adds more dimensions to a low dimensional space to make it easier to segregate the data. It converts the inseparable problem to separable problems by adding more dimensions using the kernel trick. A support vector machine is implemented in practice by a kernel. The kernel trick helps to make a more accurate classifier.

- Loading the data
- Exploring Data
- Splitting Data
- Generating The Model
- Model Evaluation

Code :

svm_aai_clg.py - C:\Users\mazhar\Desktop\svm_aai_clg.py (3.10.8)

File Edit Format Run Options Window Help

```
from sklearn import datasets


cancer_data = datasets.load_breast_cancer()
print(cancer_data.data[5])

print(cancer_data.data.shape)
print(cancer_data.target)

from sklearn.model_selection import train_test_split
cancer_data = datasets.load_breast_cancer()
x_train,x_test,y_train,y_test = train_test_split(cancer_data.data,cancer_data.target,test_size=0.4,random_state=109)

from sklearn import svm
cls = svm.SVC(kernel="linear")
cls.fit(x_train,y_train)
pred = cls.predict(x_test)

import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn import svm

letters = datasets.load_digits()
clf = svm.SVC(gamma=0.001,C=100)
X,y=letters.data[:-10],letters.target[:-10]
clf.fit(X,y)
print(clf.predict(letters.data[:-10]))
plt.imshow(letters.images[6],interpolation='nearest')
plt.show()
```

Output :

IDLE Shell 3.10.8

File Edit Shell Debug Options Window Help

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: C:\Users\mazhar\Desktop\svm_aai_clg.py ===============
[1.245e+01 1.570e+01 8.257e+01 4.771e+02 1.278e-01 1.700e-01 1.578e-01
 8.089e-02 2.087e-01 7.613e-02 3.345e-01 8.902e-01 2.217e+00 2.719e+01
 7.510e-03 3.345e-02 3.672e-02 1.137e-02 2.165e-02 5.082e-03 1.547e+01
 2.375e+01 1.034e+02 7.416e+02 1.791e-01 5.249e-01 5.355e-01 1.741e-01
 3.985e-01 1.244e-01]
(569, 30)
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0
 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1 1
 1 1 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1
 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 1 1 0 0 0 1 0
 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 1 1
 1 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 0 1 1 1 1 0 1 1 1 1 1 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1
 1 1 0 1 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0
 0 1 0 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1 1
 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1
 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0
 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 0 0 0 0 0 1]
[0 1 2 ... 5 7 9]
```

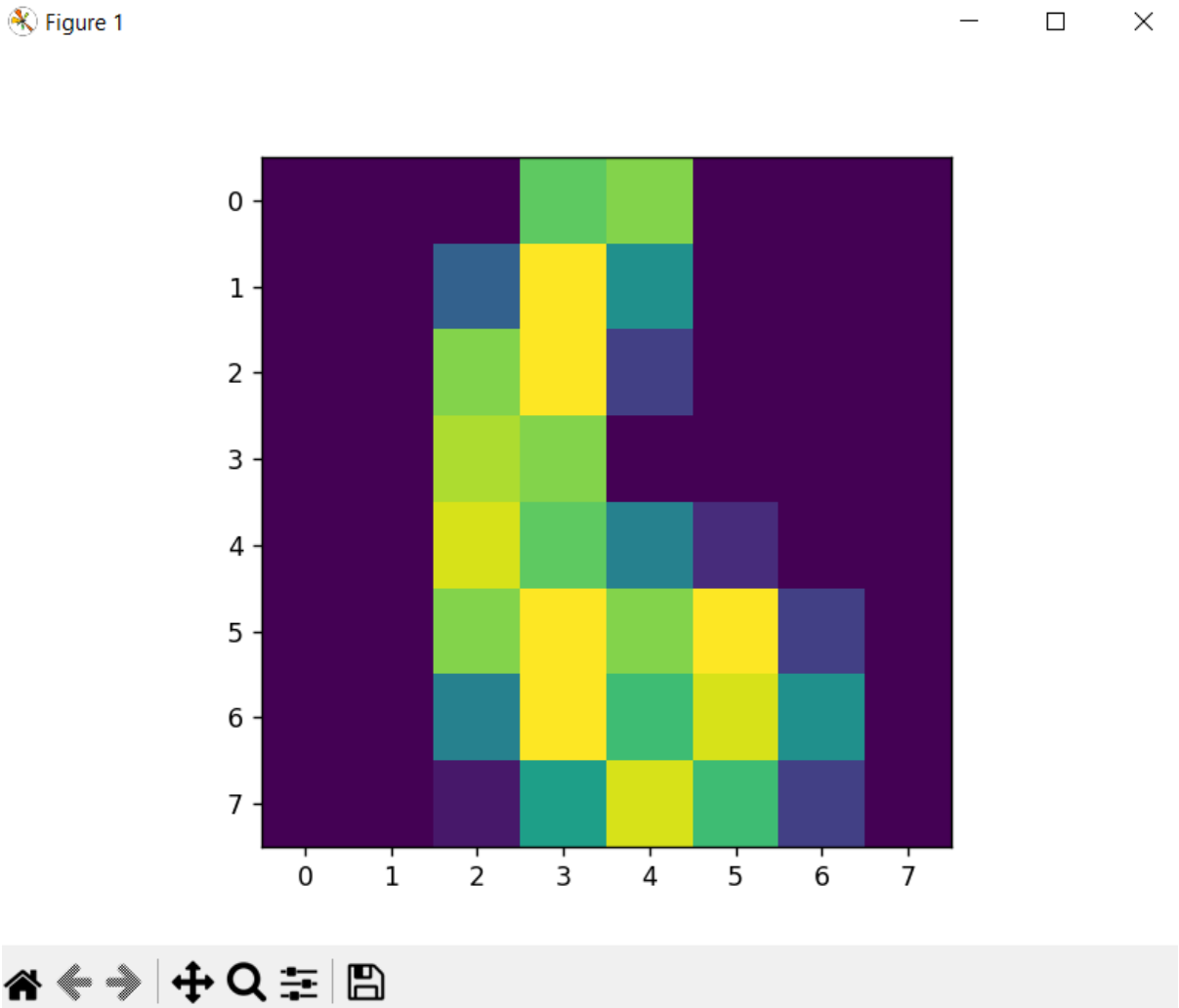


Figure 1
0
1
2
3
4
5
6
7
0
1
2
3
4
5
6
7

Practical 12

Aim : Write a program to implement parser in natural language processing

Description :

A natural language parser is a program that figures out which group of words go together (as “phrases”) and which words are the subject or object of a verb. The NLP parser separates a series of text into smaller pieces based on the grammar rules. If a sentence that cannot be parsed may have grammatical errors.

Code :

nltk_pract12.py U ×

nltk > nltk_pract12.py > ...

```
import nltk
from nltk import*
text="Today is MSc.IT practical session."
print(nltk.word_tokenize(text))
text="Today is MSc.IT practical session.It is last session for AAI."
print(nltk.sent_tokenize(text))
from nltk import pos_tag
tokens=nltk.word_tokenize(text)
pos_list=pos_tag(tokens)
```

Output :

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\MSC-IT\Applied Artificial Intelligence> & C:/Users/mazhar/AppData
tk_pract12.py"
['Today', 'is', 'MSc.IT', 'praxctical', 'session', '.']
['Today is MSc.IT praxctical session.It is last session for AAI.']
PS C:\MSC-IT\Applied Artificial Intelligence>
```