**Machine Learning**

**Assignment-1**

**Here is my GITHUB Link for my Machine Learning repository -**

<https://github.com/samanth27199/Machine-learning>

**Question 1:**

The following is a list of 10 students ages:

ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

• Sort the list and find the min and max age

• Add the min age and the max age again to the list

• Find the median age (one middle item or two middle items divided by two)

• Find the average age (sum of all items divided by their number)

• Find the range of the ages (max minus min)

**Source code 1:**

import statistics

type(ages)

ages.sort()

print(ages)

#Program to find min and max for the variable age

print(min(ages))

#Program to find min and max for the variable age

print(max(ages))

ages.append(min(ages))

print(ages)

ages.append(max(ages))

print(ages)

ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

#sorting the list to find the Median

ages.sort()

print(ages)

# printing list

print("The Sorted List is : " + str(ages))

# Median of the provided list is Given Below

# Using loop + "~" operator

midValue = len(ages) // 2

result = (ages[midValue] + ages[~midValue]) / 2

# Printing result

print("Median of list is : " + str(result))

ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

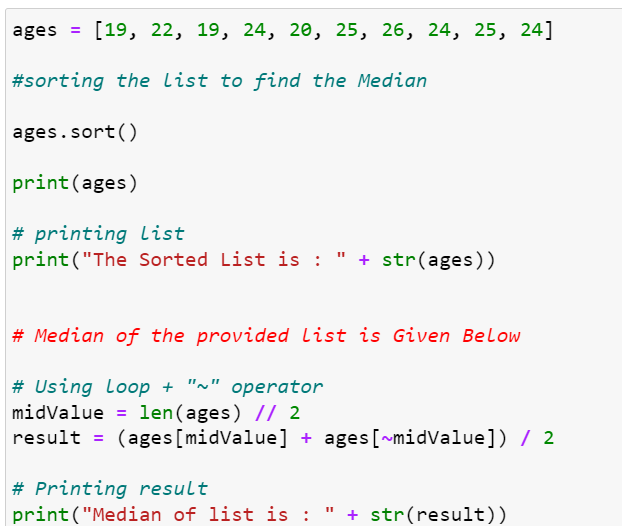
ages.sort()

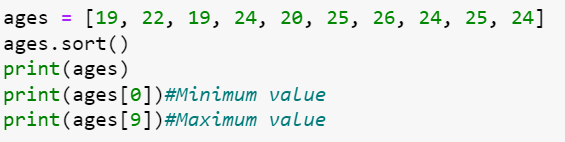
print(ages)

print(ages[0])#Minimum value

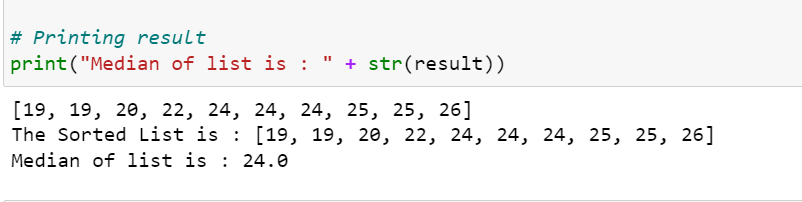
print(ages[9])#Maximum value

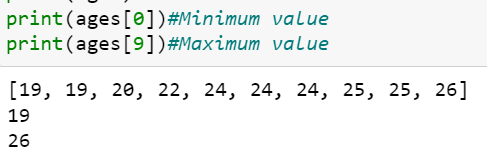
**Source code screenshot:**





**Output 1:**





**Question 2:**

• Create an empty dictionary called dog

• Add name, color, breed, legs, age to the dog dictionary

• Create a student dictionary and add first\_name, last\_name, gender, age, marital status, skills, country, city and address as keys for the dictionary

• Get the length of the student dictionary

• Get the value of skills and check the data type, it should be a list

• Modify the skills values by adding one or two skills

• Get the dictionary keys as a list

• Get the dictionary values as a list

**Source code 2:**

import statistics

# Create an empty dictionary called dog

#Empty Dictionary

dog={}

print(type(dog))

dog={'name':'Adisi','color':'Brown','Breed':'Labrador','legs':'short','age':4}

print(dog)

print(dog['name'])

student = {'first\_name':'Samanth', 'last\_name':'satvai', 'gender':'Male', 'age':'25', 'marital status':'Single','skills':'Reading', 'country':'India', 'city':'Hyderabad', 'address':'KPHB'}

print (student) #printing the student dictionary

print(len(student)) #finding the length of the student dictionary using len function

print(student['skills']) #Getting the value of skills

print(type(student['skills'])) #checking the data type of skills

student['skills']='cycling','music' #updating the skills

print(student)

keys = print(student.keys()) #getting the dictionary keys as list

values = print(student.values()) #getting the dictionary values as list

MyDict={'first\_name':'samanth', 'last\_name':'satvai' , 'gender':'Male', 'age':21, 'martial status':"Single", 'skills':['c','Python','Java','SQL'],'country':'USA', 'city': 'Overland Park', 'address':'8601 W 131st Ter Overland Prk Kansas 66213'}

MyDict['skills'].append('AWS')

MyDict['skills'].append('Java-Script')

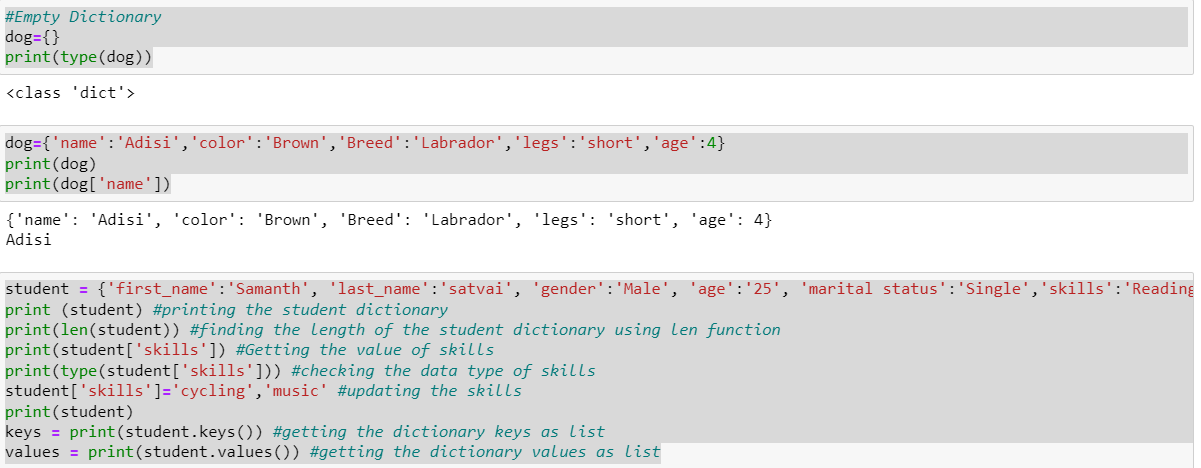
print("The Added vakues in the List are:",MyDict)

MyDict={'first\_name':'samanth', 'last\_name':'satvai' , 'gender':'Male', 'age':21, 'martial status':"Single", 'skills':['c','Python','Java','SQL'],'country':'USA', 'city': 'Overland Park', 'address':'8601 W 131st Ter Overland Prk Kansas 66213'}

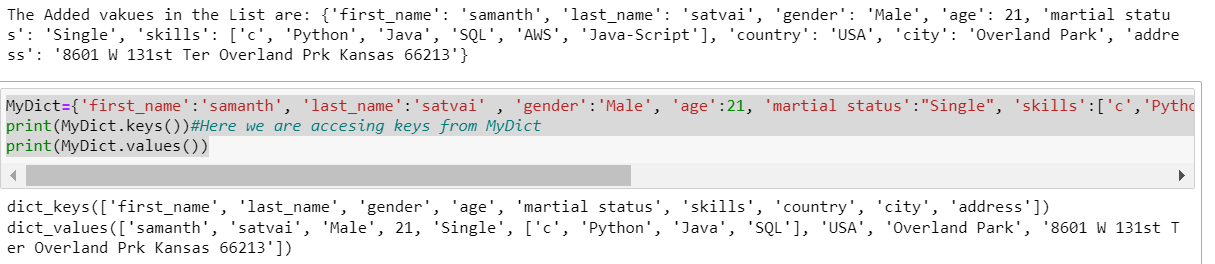
print(MyDict.keys())#Here we are accesing keys from MyDict

print(MyDict.values())

**Source code screenshot:**



**Output 2:**



**Question 3:**

* Create a tuple containing names of your sisters and your brothers (imaginary siblings are fine)
* Join brothers and sisters tuples and assign it to siblings
* How many siblings do you have?
* Modify the siblings tuple and add the name of your father and mother and assign it to family\_members

**Source code 3:**

"""Create a tuple containing names of your sisters and your brothers (imaginary siblings are

fine)"""

MySisters=('Mahitha','Suma','JayaSree')

print(type(MySisters))

print("The People in the Tuple are :",MySisters)

MyBrothers=("Hemanth","Vikranth","Chaitanya")

print("The People in the Tuple are:",MyBrothers)

sibilings = MySisters+MyBrothers

print("On a Whole my Sibilings are :",sibilings)

print("Lenght of the tuple named sibilings is :",len(sibilings))

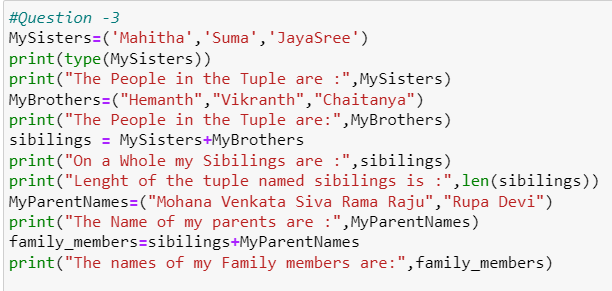
MyParentNames=("Mohana Venkata Siva Rama Raju","Rupa Devi")

print("The Name of my parents are :",MyParentNames)

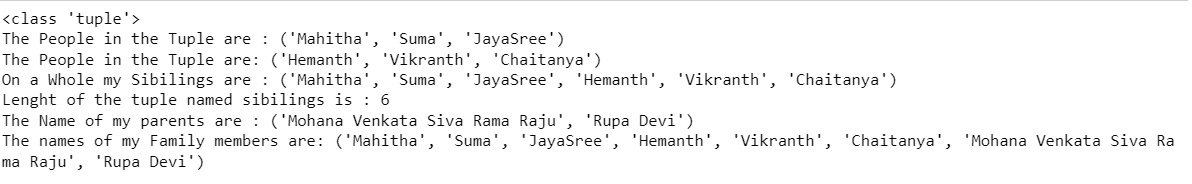
family\_members=sibilings+MyParentNames

print("The names of my Family members are:",family\_members)

**Source code screenshot:**



**Output 3:**



**Question 4:**

it\_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'}

A = {19, 22, 24, 20, 25, 26} B = {19, 22, 20, 25, 26, 24, 28, 27}

age = [22, 19, 24, 25, 26, 24, 25, 24]

• Find the length of the set it\_companies

• Add 'Twitter' to it\_companies

• Insert multiple IT companies at once to the set it\_companies

• Remove one of the companies from the set it\_companies

• What is the difference between remove and discard

• Join A and B

• Find A intersection B

• Is A subset of B

• Are A and B disjoint sets

• Join A with B and B with A

• What is the symmetric difference between A and B

• Delete the sets completely

• Convert the ages to a set and compare the length of the list and the set.

**Source code 4:**

#Question -4

it\_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'}

print(len(it\_companies))

it\_companies.add("Twitter")

print("It Companies are :",it\_companies)#Here in Set while adding the items in it, it randomly allocates its place anywhere in the set

it\_companies.update(["DBS","NCR","TCS","Infosys"])#Here in Set as add() only supports only one argument we woiuld be using update() to add multiple items

print("Names of the Multiple Company names which are added to the Original One :",it\_companies)

it\_companies.remove("Twitter")#Here for removing a particualr element we would be using remove() ,to remove a random element we would be using pop() method

A = {19, 22, 24, 20, 25, 26}

B = {19, 22, 20, 25, 26, 24, 28, 27}

Res=A.union(B)

print("Joining the sets of A and will be :",Res)#Here Union Combines all the elements in the two sets which doesnt include the duplicates in both of the sets

Res1=A.intersection(B)

print("Intersection of the sets A and B is :",Res1)#It returns the common items present in the two sets

print("Finding Whether A is Subset of B? (T/F):",A.issubset(B))#Here as all the elements in A are in B we can say that A is Subset of B,if it isnt true then issubset() return false basically it returns Boolean Values based on the Context

print("Finding Whether A and B are Disjoint sets or not?",A.isdisjoint(B))#Here it is false because there are similar items that are present in the both of the sets.The main Criteria for Disjoint Sets is that it shouldn't have common items present.

print("The Symmetric difference between the two sets A And are:",B.symmetric\_difference(A))

#It Basically caluculates{(A\B)∪(B\A)} to find Symmetric Difference

ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

print(type(ages))

print(set(ages))

print(type(ages))#List Converted to set.

#Below is the code for comparing the lenght for both list and set.

if len(ages)==len(A):

print("Both the set and List are having same no of items i.e Lenght")

else:

print("The Lenght of list and set are not same")

del A

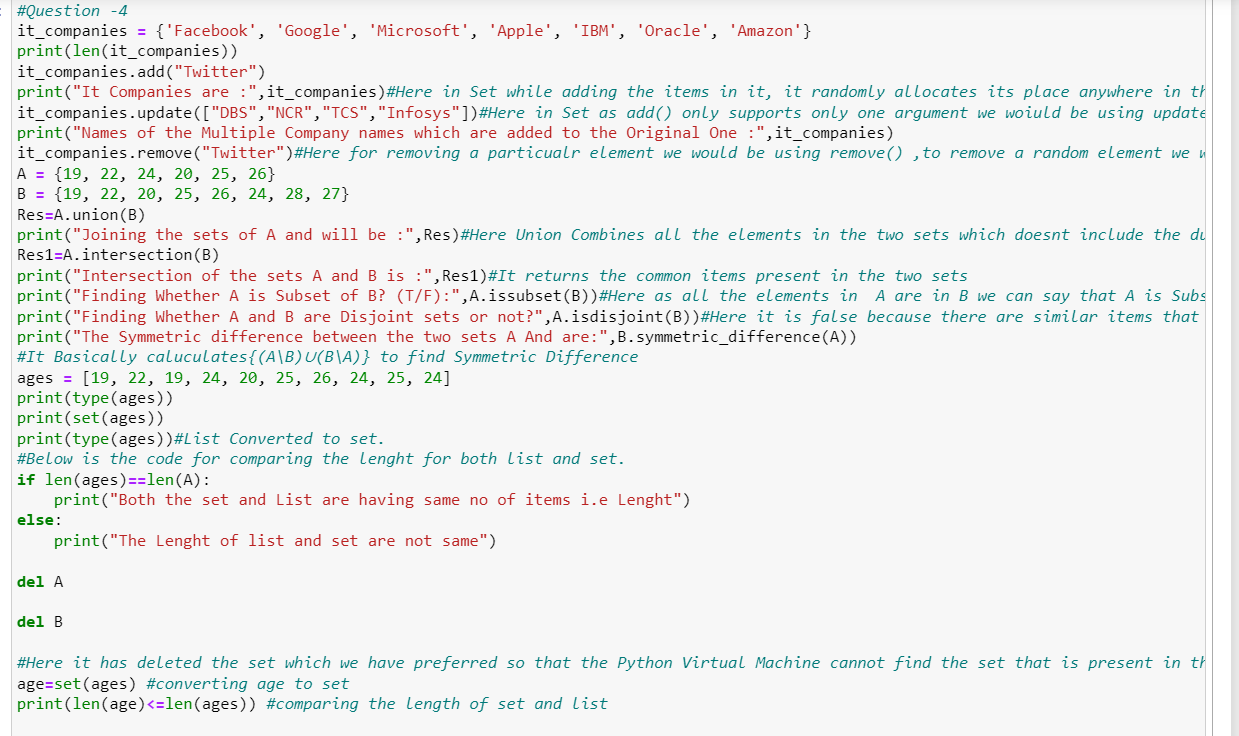
del B

#Here it has deleted the set which we have preferred so that the Python Virtual Machine cannot find the set that is present in the memory

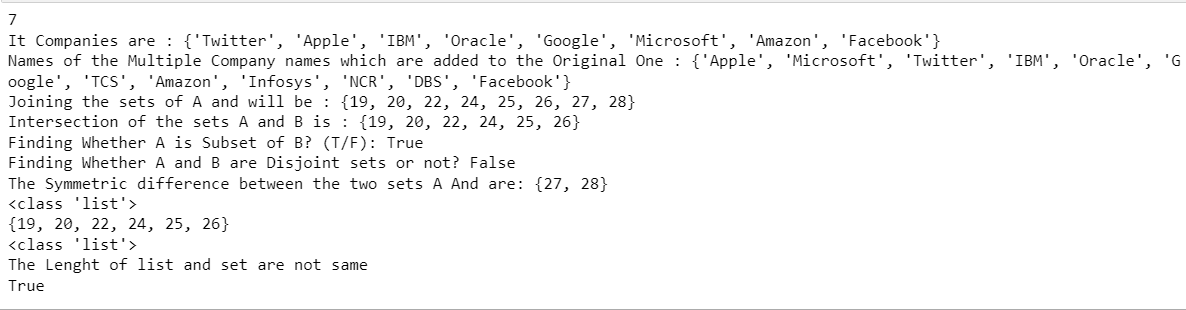
age=set(ages) #converting age to set

print(len(age)<=len(ages)) #comparing the length of set and list

**Source code screenshot:**



**Output 4:**



**Question 5:**

The radius of a circle is 30 meters.

• Calculate the area of a circle and assign the value to a variable name of \_area\_of\_circle\_

• Calculate the circumference of a circle and assign the value to a variable name of \_circum\_of\_circle\_

• Take radius as user input and calculate the area

**Source code 5:**

PI = 3.14

def AreaOfCircle(radius):

\_area\_of\_circle\_= 3.14\*radius\*radius

print("Area of Circle is:",\_area\_of\_circle\_)

def CircumferenceOfCircle(radius):

\_circum\_of\_circle\_ = 2\*3.14\*radius

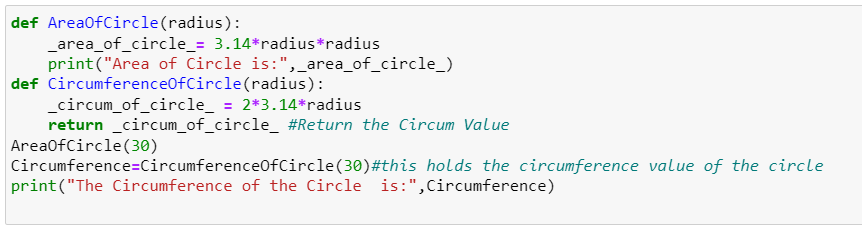
return \_circum\_of\_circle\_ #Return the Circum Value

AreaOfCircle(30)

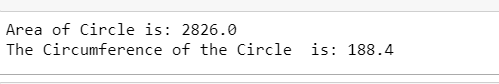
Circumference=CircumferenceOfCircle(30)#this holds the circumference value of the circle

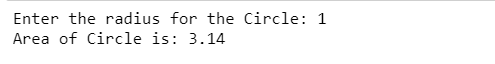
print("The Circumference of the Circle is:",Circumference)

**Source code screenshot:**



**Output 5:**





**Question 6:**

“I am a teacher and I love to inspire and teach people”

• How many unique words have been used in the sentence? Use the split methods and set to get the unique words

**Source code 6:**

#How many unique words have been used in the sentence?use the split methods and set to get the unique words

#Input=“I am a teacher and I love to inspire and teach people”

string1 = input("Please Enter the String/Sentence ")

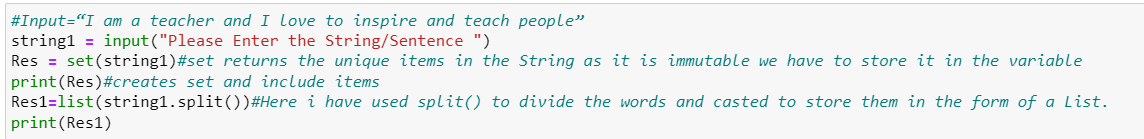
Res = set(string1 )#set returns the unique items in the String as it is immutable we have to store it in the variable

print(Res)#creates set and include items

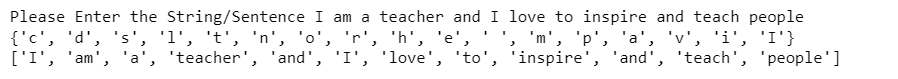
Res1=list(string1.split()) #Here i have used split() to divide the words and casted to store them in the form of a List.

print(Res1)

**Source code screenshot:**



**Output 6:**



**Question 7:**

Use a tab escape sequence to get the following lines.

Name Age Country City

Asabeneh 250 Finland Helsinki

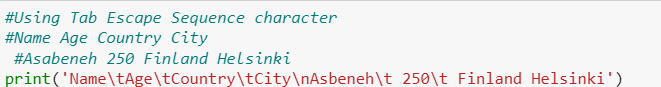
Source code 7:

#Using Tab Escape Sequence character

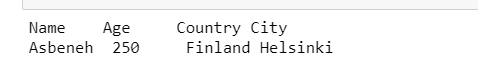
#Name Age Country City

#Asabeneh 250 Finland Helsinki

print('Name\tAge\tCountry\tCity\nAsbeneh\t 250\t Finland Helsinki')



**Output 7:**



**Question 8:**

Use the string formatting method to display the following:

radius = 10

area = 3.14 \* radius \*\* 2

“The area of a circle with radius 10 is 314 meters square.”

**Source code 8:**

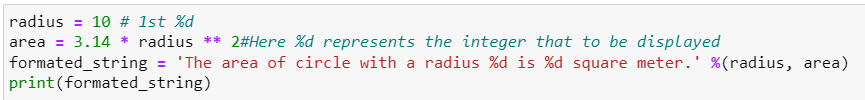
radius = 10 # 1st %d

area = 3.14 \* radius \*\* 2#Here %d represents the integer that to be displayed

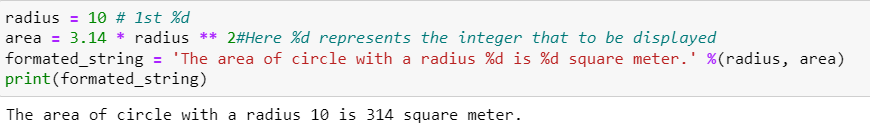
formated\_string = 'The area of circle with a radius %d is %d square meter.' %(radius, area)

print(formated\_string)

**Source code screenshot:**



**Output 8:**



**Question 9:**

Write a program, which reads weights (lbs.) of N students into a list and convert these weights to kilograms in a separate list using Loop. N: No of students (Read input from user)

Ex: L1: [150, 155, 145, 148]

Output: [68.03, 70.3, 65.77, 67.13]

**Source code 9:**

# Number of entries for students

no\_of\_students = int(input("Enter the number of students"))

wts = []

for i in range(no\_of\_students):

txt = "Enter the " + str(i) + "th weights(lbs)"

wts.append(float(input(txt)))

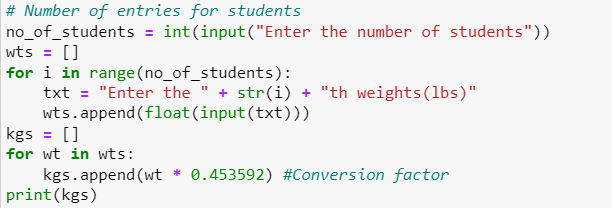
kgs = []

for wt in wts:

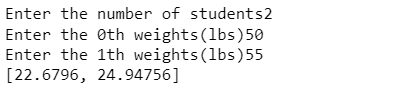
kgs.append(wt \* 0.453592) #Conversion factor

print(kgs)

**Source code screenshot:**



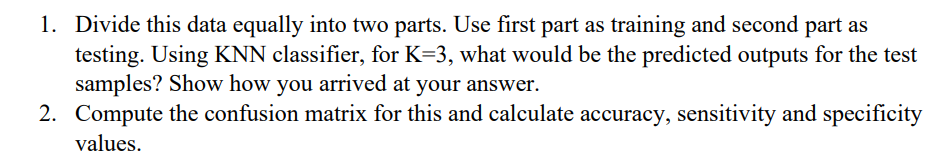
**Output 9:**



**Question 10:**

The diagram below shows a dataset with 2 classes and 8 data points, each with only one feature value, labeled f. Note that there are two data points with the same feature value of 6. These are shown as two x’s one above the other.

Chart, box and whisker chart

Description automatically generated

**Explanation:**

Given no of classes: 2 (Dots and cross)

Number of data points: 8

Dot: represented with 0

Cross: represented with 1

Mapping arrays for these two classes will be as follows:

X=np.array[1,2,3,6,6,7,10,11]

Y=np.array[0,0,1,1,1,0,0,0]

On dividing the data into two parts

1st à Training data

2ndà Testing data

For splitting the data for training and testing we use train\_test\_split method

Test\_size=0.5 (divided half for training and other half for testing)

Algorithm predicted feature value will be determined based on the constant value

In order to set the random\_state argument,seed is set to 42(all picks the same data sets for training)

On considering the samples [2,7,1,11] the nearest neighbors are calculated and are [1,0.66,1,0.66]

Confusion Matrix: It is the method which compares the actual target values with that are predicted by the machine learning mechanism.

Actual values vs Predicted Values wrt Positive and negative values

TP = cm[1][1] // [ Actual result ] [ predicted result ]

TN = cm[0][0]

FP = cm[0][1]

FN = cm[1][0]

**Accuracy:**

Accuracy is defined as the ratio of (TP + TN) with total observations.

Accuracy= (float(TP + TN) / float(TP + TN + FP + FN))

**Sensitivity:**

Sensitivity is defined as the number of correctly identified points in the class(TP) divided by the total number of positives(P)

P=TP+FN

Sensitivity=TP/P

= (TP / float(TP + FN))

**Specificity:**

Specificity is used to determine the true negative rate.

Specificity= (TN / float(TN + FP))

Text, letter

Description automatically generated

A piece of paper with writing on it

Description automatically generated