CS 5710 Machine Learning

Assignment 1

Name: Aravinda Krishna Gorantla

ID: 700741775

GitHub Link: https://github.com/gakrish5/MachineLearning/tree/main/Assignment%201

Video Link:

https://drive.google.com/file/d/11dO49bbLgZLh0B2PUmxh0L5CyyyIXZZ1/view?usp=share_link

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:

```
#To find the median of the list, importing the statistics module
import statistics

#declared a list with 10 students ages
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
print("declared ages list:",ages)

#to sort the ages list sort() function is used
ages.sort()
print("Sorted ages list:", ages)

#Finding Min age of the ages list
minimum_age = min(ages)
print("Minimum Age:", minimum_age)

#Finding Max age of the ages list
maximum_age = max(ages)
print("Maximum Age:", maximum_age)
print("----------------------------------------------------------------------------------------------------")

#adding min and max age again to the ages list using append() function
ages.append(minimum_age)
ages.append(maximum_age)
print("After adding min and max age, the ages list:", ages)
print("----------------------------------------------------------------------------------------------------")

print("Median of the ages list:", statistics.median(ages)) #finding the median of the ages list
print("----------------------------------------------------------------------------------------------------")
print("Average of the ages list:", sum(ages)/len(ages)) #finding the averages of the ages list
print("----------------------------------------------------------------------------------------------------")
print("Range of the ages list:", max(ages)-min(ages)) #finding the range of ages list
```

Output:

```
declared ages list: [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
Sorted ages list: [19, 19, 20, 22, 24, 24, 24, 25, 25, 26]
Minimum Age: 19
Maximum Age: 26
-----------------------------------------------------------------------------------------------
After adding min and max age, the ages list: [19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 19, 26]
-----------------------------------------------------------------------------------------------
Median of the ages list: 24.0
-----------------------------------------------------------------------------------------------
Average of the ages list: 22.75
-----------------------------------------------------------------------------------------------
Range of the ages list: 7
```

Explanation:

Here in the code, I have initially declared a list named "ages" with the ages of 10 students and sorted the list using the ***sort***() function.

$$sort(ages)$$

The functions ***min***() and ***max***() are used to find Minimum and the maximum age in the list.

$$max(ages), min(ages)$$

append() function is used to add the minimum and the maximum age again to the list.

$$ages.append(max(ages))$$

$$ages.append(min(ages))$$

The ***median***() function of ***Statistics*** module is used to find the median of the list. The ***Statistics*** module is imported.

$$statistics.median(ages)$$

The average of list is shown by dividing sum of all ages by total number of ages in the list. To find the average of the list, functions ***sum***() and ***len***() are used. ***sum***() function gives the sum of all values in the list and ***len***() function gives the length of the list which determines the capacity (total number) of the list.

$$average = sum(ages)/len(ages)$$

The range of the list is found by making the difference between maximum and minimum age in the list.

$$Range = max(ages) - \min(ages)$$

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:

```
dog = {} #creating an empty dictionary named as dog
print(type(dog)) #printing the data type of dog variable
#assigning key-value pairs to the dog dictionary
dog["name"] = "Rocky"
dog["color"] = "Gold"
dog["breed"] = "Golden Retriever"
dog["legs"] = 4
dog["age"] = 1

print(dog) #printing the dog dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
print("dog dictionary keys as a list:\n", dog.keys()) #printing the keys of dog dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
print("dog dictionary values as a list:\n", dog.values()) #printing the values of dog dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
```

```
#creating a dictionary named as student and assigning key-value pairs
student = {'first_name': 'Aravinda Krishna',
           'last_name': 'Gorantla',
           'gender': 'Male',
           'age': 26,
           'marital_status': 'Unmarried',
           'skills': ['Python', 'MongoDB', 'Neo4j', 'SQL'],
           'country': 'United States',
           'city': 'Overland Park',
           'address': {'street': '11250 Glenwood St', 'Zipcode': 66211}}

print(student) #printing the student dictionary

print("-----------------------------------------------------------------------------------------------------------------------")
print("Length of student dictionary:", len(student)) #printing the length of the student dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
print(student["skills"]) #printing values of skills
print(type(student["skills"])) #printing the data type of skills in the dictionary
print("-----------------------------------------------------------------------------------------------------------------------")

#adding new skills to the skills list
student["skills"].append("Java")
student["skills"].append("Machine Learning")
print(student["skills"])

print("-----------------------------------------------------------------------------------------------------------------------")
print("Student dictionary keys as a list:\n", student.keys()) #printing the keys of student dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
print("Student dictionary values as a list:\n", student.values()) #printing the values of student dictionary
print("-----------------------------------------------------------------------------------------------------------------------")
```

Output:

```
<class 'dict'>
{'name': 'Rocky', 'color': 'Gold', 'breed': 'Golden Retriever', 'legs': 4, 'age': 1}
------------------------------------------------------------------------------------------------------------------------
dog dictionary keys as a list:
 dict_keys(['name', 'color', 'breed', 'legs', 'age'])
------------------------------------------------------------------------------------------------------------------------
dog dictionary values as a list:
 dict_values(['Rocky', 'Gold', 'Golden Retriever', 4, 1])
------------------------------------------------------------------------------------------------------------------------
{'first_name': 'Aravinda Krishna', 'last_name': 'Gorantla', 'gender': 'Male', 'age': 26, 'marital_status': 'Unmarried', 'skill
s': ['Python', 'MongoDB', 'Neo4j', 'SQL'], 'country': 'United States', 'city': 'Overland Park', 'address': {'street': '11250 Gl
enwood St', 'Zipcode': 66211}}
------------------------------------------------------------------------------------------------------------------------
Length of student dictionary: 9
------------------------------------------------------------------------------------------------------------------------
['Python', 'MongoDB', 'Neo4j', 'SQL']
<class 'list'>
------------------------------------------------------------------------------------------------------------------------
['Python', 'MongoDB', 'Neo4j', 'SQL', 'Java', 'Machine Learning']
------------------------------------------------------------------------------------------------------------------------
Student dictionary keys as a list:
 dict_keys(['first_name', 'last_name', 'gender', 'age', 'marital_status', 'skills', 'country', 'city', 'address'])
------------------------------------------------------------------------------------------------------------------------
Student dictionary values as a list:
 dict_values(['Aravinda Krishna', 'Gorantla', 'Male', 26, 'Unmarried', ['Python', 'MongoDB', 'Neo4j', 'SQL', 'Java', 'Machine L
earning'], 'United States', 'Overland Park', {'street': '11250 Glenwood St', 'Zipcode': 66211}])
------------------------------------------------------------------------------------------------------------------------
```

Explanation:

Here in the code, I have declared an empty dictionary named “dog” and the key value pairs are assigned as below.

$$dog = \{\}$$
$$dog["name"] = "Rocky"$$
$$dog["color"] = "Gold"$$
$$dog["breed"] = "Golden\ Retriever"$$
$$dog["legs"] = 4$$
$$dog["age"] = 1$$

The dictionary dog is printed with its keys and values.

$$\{'name': 'Rocky', 'color': 'Gold', 'breed': 'Golden\ Retriever', 'legs': 4, 'age': 1\}$$

A dictionary named “student” is declared which includes the fields such as first_name, last_name, age, marital status, skills, country, city, and address as keys and its values are also assigned as shown below.

$$\begin{aligned} student = \{&'first\_name': 'Aravinda\ Krishna',\\ &'last\_name': 'Gorantla',\\ &'gender': 'Male',\\ &'age': 26,\\ &'marital\_status': 'Unmarried',\\ &'skills': ['Python', 'MongoDB', 'Neo4j', 'SQL'],\\ &'country': 'United\ States',\\ &'city': 'Overland\ Park',\\ &'address': \{'street': '11250\ Glenwood\ St', 'Zipcode': 66211\}\} \end{aligned}$$

The dictionary student is printed with its keys and values.

$\{'first\_name': 'Aravinda\ Krishna', 'last\_name': 'Gorantla', 'gender': 'Male', 'age': 26,$

$'marital\_status': 'Unmarried', 'skills': ['Python', 'MongoDB', 'Neo4j', 'SQL'],$

$'country': 'United\ States', 'city': 'Overland\ Park',$

$'address': \{'street': '11250\ Glenwood\ St', 'Zipcode': 66211\}\}$

The length of student dictionary is found using the function $len()$.

$$len(student)$$

The values of skills are printed accessing the skills from student dictionary.

$$student["skills"]$$

The data type of skills has been found using the $type()$ function.

$$type(student["skills"])$$

The Skills list is modified by adding new skills “Java” and “Machine Learning” to the list using the $append()$ function.

$$student["skills"].append("Java")$$

$$student["skills"].append("Machine\ Learning")$$

The dictionary keys are printed using the $keys()$ function.

$$dog.keys()$$

$$student.keys()$$

The dictionary values are printed using the $values()$ function.

$$dog.values()$$

$$student.values()$$

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 sibling's tuple and add the name of your father and mother and assign it to family_members

Source code:

```
sisters = ("Siri","Chinni","Chinnari") #sisters tuple
brothers = ("Sankar","Sumanth","Ruthvik","Karthik") #brothers tuple
siblings = brothers + sisters #Joining the tuples sisters and brothers
print(siblings)
print("-----------------------------------------------------------------------------------------------------------------------")
print("No of Siblings: ", len(siblings)) #printing the length of the siblings tuple
print("-----------------------------------------------------------------------------------------------------------------------")
siblings_list = list(siblings) #converting the siblings tuple to list

#Modifying the siblings list by appending father and mother name
siblings_list.append("Siva Sankara Rao")
siblings_list.append("Hema Lakshmi")
family_members = tuple(siblings_list) #converting the list into tuple
print(family_members)
```

Output:

```
('Sankar', 'Sumanth', 'Ruthvik', 'Karthik', 'Siri', 'Chinni', 'Chinnari')
-----------------------------------------------------------------------------------------------------------------------
No of Siblings:  7
-----------------------------------------------------------------------------------------------------------------------
('Sankar', 'Sumanth', 'Ruthvik', 'Karthik', 'Siri', 'Chinni', 'Chinnari', 'Siva Sankara Rao', 'Hema Lakshmi')
```

Explanation:

Here in this code, I have created two tuples named sisters and brothers with some imaginary values. Then joined both the tuples and created a new tuple named siblings.

Now converted the tuple into a list and then with the help of $append()$ function in lists, I added the father and mother name. Then converted the list back to tuple and named it as family_members.

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:

```
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]
print("Length of the set it_companies: ",len(it_companies)) #Printing the length of the set: it_companies
print("-------------------------------------------------------------------------------------------------")
it_companies.add("Twitter") #adding twitter to the set: it_companies
print(it_companies)
print("-------------------------------------------------------------------------------------------------")
it_companies.update(["Alpha","Infosys","Tesla"]) #adding multiple companies to the set: it_companies
print(it_companies)
print("-------------------------------------------------------------------------------------------------")
it_companies.remove("Amazon") #removing the company Amazon from the set: it_companies
print(it_companies)

'''it_companies.remove("Amazon")''' #When we try to remove Amazon again, we will get an error as it is not present in the set
it_companies.discard("Amazon") #if we use discard, then we dont get any error even if it is not present in the set

'''remove() raises an error when the element not exists in the set. where as discard() does not raise any error
even if the element is not found in the set'''
print("-------------------------------------------------------------------------------------------------")
```

```
C = A | B #performs the union of A and B sets
print("Union of A and B: ",C)
print("----------------------------------------------------------------------------------------------------")
D = A & B #performs the intersection of A and B sets
print("Intersection of A and B: ",D)
print("----------------------------------------------------------------------------------------------------")
print("Is A subset of B: ",A.issubset(B)) #checking if A is subset of B
print("----------------------------------------------------------------------------------------------------")
print("Are A and B disjoint sets: ",A.isdisjoint(B)) #checking whether the intersection of A and B is null or not
print("----------------------------------------------------------------------------------------------------")
print("After joining A with B: ",A.union(B)) #joining A with B
print("After joining B with A: ",B.union(A)) #joining B with A
print("----------------------------------------------------------------------------------------------------")
print("Symmetric difference of A and B: ",A.symmetric_difference(B)) #printing the compliment of A intersection B
print("----------------------------------------------------------------------------------------------------")
del it_companies #deleting the set it_companies
del A  #deleting the set A
del B  #deleting the set B
del C  #deleting the set C
del D  #deleting the set D

age_set = set(age) #converting the age list into a set
print("Length of list age: ",len(age)) #printing the length of the age list
print("Length of age_set is: ",len(age_set)) #printing the length of age_set
print("----------------------------------------------------------------------------------------------------")
```

Output:

```
Length of the set it_companies:  7
----------------------------------------------------------------------------------------------------
{'Facebook', 'IBM', 'Apple', 'Oracle', 'Twitter', 'Google', 'Microsoft', 'Amazon'}
----------------------------------------------------------------------------------------------------
{'Facebook', 'Infosys', 'Tesla', 'Amazon', 'Oracle', 'IBM', 'Alpha', 'Apple', 'Twitter', 'Google', 'Microsoft'}
----------------------------------------------------------------------------------------------------
{'Facebook', 'Infosys', 'Tesla', 'Oracle', 'IBM', 'Alpha', 'Apple', 'Twitter', 'Google', 'Microsoft'}
----------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------
Union of A and B:  {19, 20, 22, 24, 25, 26, 27, 28}
----------------------------------------------------------------------------------------------------
Intersection of A and B:  {19, 20, 22, 24, 25, 26}
----------------------------------------------------------------------------------------------------
Is A subset of B:  True
----------------------------------------------------------------------------------------------------
Are A and B disjoint sets:  False
----------------------------------------------------------------------------------------------------
After joining A with B:  {19, 20, 22, 24, 25, 26, 27, 28}
After joining B with A:  {19, 20, 22, 24, 25, 26, 27, 28}
----------------------------------------------------------------------------------------------------
Symmetric difference of A and B:  {27, 28}
----------------------------------------------------------------------------------------------------
Length of list age:  8
Length of age_set is:  5
----------------------------------------------------------------------------------------------------
```

Explanation:

Here in the code, I have declared a set named it_companies and its Length is found using ***len***() function. By using ***add***() function, couple of elements are added one by one to the set it_companies and printed the new set. Then we added multiple values into the set at once using ***update***() function and printed the updated set.

Using the ***remove***() function removed a value and the resultant set is printed.
Difference between $remove()$ *and* $discard()$ *functions:*
$remove()$ function raises an error when the element does not exist in the set.
whereas $discard()$ function do not raise any error even if the element is not found in the set.

Now, I have declared two sets A and B with some values.
Set functions stated below are performed and printed the results.

union() – Joins 2 sets. Can be performed using $A \mid B$

intersection() – the result set consists of common elements from both sets. Can be performed using A & B

issubset() – checks whether a set is subset of another

isdisjoint() – checks whether the intersection of A and B is a null or not

symmetric_difference() – gives the complement of A intersection B.

'**del'** is used to delete the sets.

Now a list age is declared. Typecasting is done, i.e., converting the list into a set named age_set. The length of the list and set are calculated. There is a difference in both the length values as the set does not allows duplicates.

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:

```
radius = 30 #initialising the radius of circle as 30

#calculating the area of circle using the formula pie*r*r
_area_of_circle_ = 3.14*(radius**2)
print("Area of circle with radius 30 is: ", _area_of_circle_)
#calculating the circumfrence of circle using the formula 2*pie*r
_circum_of_circle_ = 2*3.14*radius
print("Circumfrence of a circle with radius 30 is: ", _circum_of_circle_)

#taking the radius as input from user
rad = float(input("Enter radius: "))
#calculating the area of circle for the given radius
area = 3.14*(rad**2)
print("Area of circle with radius", rad, " is: ", area)
```

Output:

```
Area of circle with radius 30 is:  2826.0
Circumfrence of a circle with radius 30 is:  188.4
Enter radius: 25
Area of circle with radius 25.0  is:  1962.5
```

Explanation:

Here in the code, I have declared radius of the circle and initialized to 30.
With that radius, area and circumference of the circle are determined and printed.

Now radius is taken as the input from the user using $input()$ function. With the user provided radius, area is calculated and printed.

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:

```
string = "I am a teacher and I love to inspire and teach people" #given sentence
#splitting the given sentence using the space as a delimiter
words = string.split(" ")

#eliminating the duplicate words by converting it into set
words_set = set(words)

print("Number of unique words from the sentence is: ",len(words_set)) #printing the length of set with unique words
print("Unique words from the sentence is: ",words_set) #printing the set of unique words
```

Output:

```
Number of unique words from the sentence is:  10
Unique words from the sentence is:  {'inspire', 'I', 'teacher', 'to', 'people', 'and', 'a', 'am', 'love', 'teach'}
```

Explanation:

A variable named string is declared and initialized with the given text. The string is made into words as stored as a list using $split()$ function and space as the delimiter. The list is typecasted as set because set will not allow duplicates.

Now the length of the set gives number of unique words and set gives the unique words. Both the unique words and its count are printed.

Question 7:

Use a tab escape sequence to get the following lines.

Name	**Age**	**Country**	**City**
Asabeneh	**250**	**Finland**	**Helsinki**

Source code:

```
# '\' is an escape character and \t gives a tab space, \n gives a new line
#printing in a single print statement by using the tab space, new line
print("Name\t\tAge\tCountry\t\tCity\nAsabeneh\t250\tFinland\t\tHelsinki")
```

Output:

```
Name            Age     Country         City
Asabeneh        250     Finland         Helsinki
```

Explanation:

Both the lines given are printed in the same pattern using the escape tab sequence "\t" which gives 3 space characters and "\n" used to get a new line.

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:

```
radius = 10 #initializing radius to 10
area = 3.14*radius**2 #calculating the area

#printing the result with text
print("The area of a circle with radius {} is {} meters square".format(radius,int(area)))
```

Output:

```
The area of a circle with radius 10 is 314 meters square
```

Explanation:

Here in the code, radius is initialized and area is calculated. The $format()$ method formats the specified value(s) and inserts them inside the string's placeholder.

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:

```
a = N = int(input("Enter number of Students:")) #taking number of weights as N
lbs_list = [] #empty list to store weights in lbs
i=1

#while loop is used to get all the weights in lbs from user
while(N != 0):
    m = int(input("Enter the Weight of Student {} in lbs: ".format(i))) #input of weights from user
    lbs_list.append(m) #appending each weight to the lbs_list
    N = N - 1
    i = i + 1
print("\nWeights of {} Students in lbs: ".format(a), lbs_list)

kgs_list = [] #empty list to store weights in kgs
N = len(lbs_list)
j = 0
while(j != N): #while loop helps us to iterate through lbs_list
    kgs_list.append(lbs_list[j]*0.45359) #converts the weights from lbs to kgs and appends to kgs_list
    j = j + 1
print("Weights of {} Students in Kgs: ".format(a), kgs_list)
```

Output:

```
Enter number of Students:3
Enter the Weight of Student 1 in lbs: 35
Enter the Weight of Student 2 in lbs: 45
Enter the Weight of Student 3 in lbs: 55

Weights of 3 Students in lbs:  [35, 45, 55]
Weights of 3 Students in Kgs:  [15.87565, 20.41155, 24.94745]
```

Explanation:

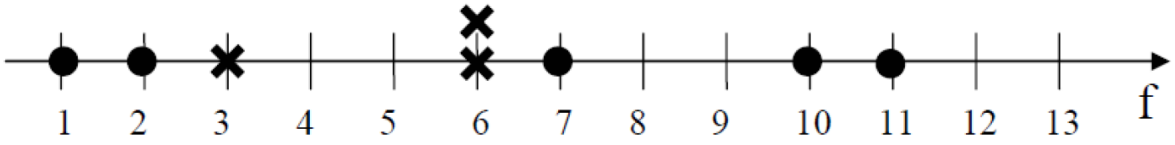
Here in the code, user input is taken for the number of students and an empty list is created. By using while loop, the individual weights of students in lbs. is taken as input and added to the empty list.

Another empty list is created. By using while loop, the individual weights of students in lbs. converted **to kgs.** is done and added to the new list, the list of converted values is printed.

Conversion formulae: $1\ kg = 0.45359\ lbs$

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. Provide stepwise mathematical solution, do not write code for it.



1. Divide this data equally into two parts. Use first part as training and second part as testing. Using KNN classifier, for $K = 3$, what would be the predicted outputs for the test samples? Show how you arrived at your answer.
2. Compute the confusion matrix for this and calculate accuracy, sensitivity, and specificity values.

Solution:

Given data elements from the diagram are taken in the tabular form as below.

Feature	Label
1	O
2	O
3	X
6	X
6	X
7	O
10	O
11	O

The given dataset is divided equally and first four rows of data in the table are considered to be the Training dataset and the next four rows are selected as the Testing dataset.

Using KNN Classifier considering $K = 3$, the distance between the testing and training data is demonstrated below.

In KNN, we are going to find the Euclidean distance between the data points.

$$d = \sqrt{(x_1 - x_2)^2} \qquad \text{(For a single feature)}$$

The Euclidean distance between the data points is calculated and placed in the below table, where the columns are the training dataset and rows are the testing dataset.

The highlighted rows (Made bold) are the distanced values.

	1(O)	2(O)	3(X)	6(X)
6	5	**4**	**3**	**0**
7	6	**5**	**4**	**1**
10	9	**8**	**7**	**4**
11	10	**9**	**8**	**5**

Let us assume 'O' as negative and 'X' as the positive values.

Confusion matrix for a prediction would be:

TN	FP
FN	TP

True Positive (TP): When the actual label is positive (X) and your machine learning model also predicts that label as positive (X).

True Negative (TN): When the actual label is negative (O) and your machine learning model also predicts that label as negative (O).

False Positive (FP): When the actual label is negative (O), but the machine learning model predicts that label as positive (X).

False Negative (FN): When the actual label is positive (X), but the machine learning model predicts that label as negative (O).

The prediction on testing data is below.

	True label	Predicted label	O/P
6	X	X	TP
7	O	X	FP
10	O	X	FP
11	O	X	FP

The final Confusion matrix for the above prediction is:

0	3
0	1

Accuracy of the classifier $= \frac{(TP + TN)}{(P + N)} = \frac{1}{4} = 0.25$

Sensitivity of the classifier $= \frac{TP}{TP+FN} = \frac{TP}{P} = \frac{1}{1} = 1$

Specificity of the classifier $= \frac{TN}{FP+TN} = \frac{TN}{N} = \frac{0}{3} = 0$

- - - - End - - - -