Machine Learning Assignment-1

#700742289

GitHub Link : https://github.com/Sushmitha-Virri/MLAssignments21627.git

Video Link :
https://drive.google.com/file/d/1t7sIYFFs0HtnsOMpceWcKyv990Vl70YB/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)

```
# given list of ages
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
print("ages before sorting: ",ages)

# sorting the list of ages in ascending order
ages.sort()
print("ages after sorting: ",ages)

# finding minimum and maximun values for the list of ages
min_value = min(ages)
max_value = max(ages)
print("minimum age is: ",min_value)
print("maximum age is: ",max_value)

# adding minimum and maximum values to the ages list
ages.insert(0,min_value) # to make the list appear sorted we add the min value at the start of the list by using insert function
ages.append(max_value) # using the append function we add the max value at the end of the list
print("ages after adding the min and max values: ",ages)

# finding the median age
mid_index = int(len(ages)/2) # finding the middle index of the list
median_age = (ages[mid_index] + ages[len(ages)-mid_index-1]) / 2 # Calculating the median by considering the items of one index i
print("median age in the given list of ages: ",median_age)

# Finding the average age
n = len(ages) # finding the length of ages
sum_ages = sum(ages) # finding the sum of ages
average_age = sum_ages/n
print("average age in the list of ages: ",average_age)

#Finding the range of the ages
print("range of the ages list: ", max_value-min_value)
```

```
ages before sorting:  [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
ages after sorting:  [19, 19, 20, 22, 24, 24, 24, 25, 25, 26]
minimum age is:  19
maximum age is:  26
ages after adding the min and max values:  [19, 19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 26]
median age in the given list of ages:  24.0
average age in the list of ages:  22.75
range of the ages list:  7
```

To sort the ages list I have used sort() method which sorts the list in ascending order.

Then, I have used min(), max() methods to find the maximum and minimum values in the list.

To add the min value I have used insert() method to insert at 0^{th} index and then to add the max value I have used append() method to insert at last of the list.

Next, to find the median age I have first found the middle index using len() function

Then, I took average of the two middle values (i.e. indices mid_index, len(ages)-mid_index-1) which gives same value if the length is even else two different values.

Next, I found the average of the list by taking sum() method to find the sum list and the divided by the length of the list using len() function.

Next, to get the range of the list, I have subtracted max and min values of the list.

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

```python
dog={} # creating empty dictonary dog

#Add name, color, breed, legs, age to the dog dictionary
dog["name"] = "Cooper"
dog["color"] = "White"
dog["breed"] = "Samoyed"
dog["legs"] = 4
dog["age"] = 5
print("Dog dictonary: ",dog)

# Creating a student dictionary and add first_name, last_name, gender, age, marital status,
# skills, country, city and address as keys for the dictionary
student ={
    "first_name": "Virri",
    "last_name": "Sushmitha",
    "gender": "Female",
    "age": 26,
    "marital status": "married",
    "skills": ["C", "Java", "Python", "SQL"],
    "country": "USA",
    "city": "Kansas City",
    "Address": "H.No: 2/3/12, ABC Apartment, Kansas City"
}
print("\nStudent Dictonary: ", student)

# length of the student dictionary
print("\nLength of the student dictionary: ", len(student))

#get the value of the skills key in the student dictonary
print("\nSkills are: ", student["skills"])

#get the data type of the skills in the student dictonary
print("\nData Type of the skills: ",type(student["skills"])) #list
```

```python
# Modify the skills values by adding one or two skills
student["skills"].extend(["Advanced Data Structures", "C++"]) # adding two more values to the skills key using extend() method
print("\nSkills after modifying: ", student["skills"])

# Get the dictionary keys as a list
student_keys = student.keys() # using in-built keys() method to get all keys from the dictonary.
print("\nStudent dictonary keys: ",student_keys )

# Get the dictonary values as a list
student_values = student.values() # using in-built values() method to get all values from the dictonary.
print("\nStudent dictonary values: ", student_values)
```

```
Dog dictonary:  {'name': 'Cooper', 'color': 'White', 'breed': 'Samoyed', 'legs': 4, 'age': 5}

Student Dictonary:  {'first_name': 'Virri', 'last_name': 'Sushmitha', 'gender': 'Female', 'age': 26, 'marital status': 'marrie
d', 'skills': ['C', 'Java', 'Python', 'SQL'], 'country': 'USA', 'city': 'Kansas City', 'Address': 'H.No: 2/3/12, ABC Apartment,
Kansas City'}

Length of the student dictionary:  9

Skills are:  ['C', 'Java', 'Python', 'SQL']

Data Type of the skills:  <class 'list'>

Skills after modifying:  ['C', 'Java', 'Python', 'SQL', 'Advanced Data Structures', 'C++']

Student dictonary keys:  dict_keys(['first_name', 'last_name', 'gender', 'age', 'marital status', 'skills', 'country', 'city',
'Address'])

Student dictonary values:  dict_values(['Virri', 'Sushmitha', 'Female', 26, 'married', ['C', 'Java', 'Python', 'SQL', 'Advanced
Data Structures', 'C++'], 'USA', 'Kansas City', 'H.No: 2/3/12, ABC Apartment, Kansas City'])
```

First, I have created empty dictionary using {}

Then, I have added key value pairs as mentioned in the question

Then, I have created student dictionary using keys first_name, last_name, gender, age, marital status, skills, country, city and address with respective values.

To get the length of the dictionary I have used len() function

To get the skills value in the dictionary I have used student[“skills”].

To get the type of the skills value I have used type() method

In order to modify the skills values (which is a list). I have used extend method to add multiple values to it.

Lastly, to get the dictionary keys and values as list, I have used keys() and 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 siblings tuple and add the name of your father and mother and assign it to family_members

```
#Creating a tuple containing names of your sisters and your brothers
sisters = ("Mary","Jenni")
brothers = ("John", "Mark")

#joining sisters and brothers tuple
siblings = sisters+ brothers
print("Siblings tuple: ",siblings)

#Total siblings
print("\nTotal number of siblings: ",len(siblings))

#Modify the siblings tuple and add the name of your father and mother and assign it to family_member
y = list(siblings) #converting siblings tuple to list as we can't directly modify tuple (as it is immutable)
y.append("Oliver") #adding father name
y.append("Emma")#adding mother name
family_member = tuple(y) #converting list to tuple
print("\nFamily members tuple:",family_member)
```

```
Siblings tuple:  ('Mary', 'Jenni', 'John', 'Mark')

Total number of siblings:  4

Family members tuple: ('Mary', 'Jenni', 'John', 'Mark', 'Oliver', 'Emma')
```

First, I have created tuple of sisters with values “Mary” and “Jenni”

Similarly, I have created tuple of brothers with values “John” and “Mark”

Then, to join brothers and sisters tuple I have concatenated them by using “+”

To find the total no.of siblings I have used len() function.

In order to modify siblings tuple and add father and mother name, I have first converted the siblings tuple to list using list() constructor, as tuples are immutable. Then, I have added the father and mother name to list and then converted list to tuple using tuple() constructor to get the family_members tuple.

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.

```
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 it_companies: ",len(it_companies)) #Finding length of the set

it_companies.add('Twitter') # adding twitter into the set
print("\nAfter adding twitter into the set: ",it_companies) #set after adding the twitter

it_companies.update({'Cisco', 'Tesla','Samsung','Walmart Inc.'}) #inserting multiple company names to  the set
print("\nAfter adding multiple companies into the set: ",it_companies) #set after adding companies

it_companies.remove("Cisco") #removing item from set using remove function
print("\nAfter removing Cisco from the set: ",it_companies)

#Difference between discard and remove functions
#discard function will not give any error when we remove item which is not in the set whereas
# if we try to remove item which is not in the set using remove function it gives error.

#Removing item which is not in the set using discard function
it_companies.discard("Cisco") # No error

#Removing item which is not in the set using remove function
#it_companies.remove("Cisco") # KeyError: 'Cisco'

#Join A and B sets
S =  A.union(B)
print("\nSet after joining two sets A and B:",S)

# Intersection of two sets A and B
I = A.intersection(B)
print("\nIntersection of sets A and B:",I)

#Is A subset of B
print("\nChecking if A is subset of B: ", A.issubset(B))
```

```
#Check if A and B are disjoint sets
print("\nChecking if A and B are disjoint sets: ", A.isdisjoint(B))

#Join A with B and B with A
S = A.union(B)
print("\nJoining A with B:",S )

S = B.union(A)
print("\nJoining B with A:",S)

#symmetric difference between A and B
print( "\nSynmmetric difference between A and B: ",A.symmetric_difference(B))

#Delete the sets completely
A.clear()
B.clear()
print("\nA and B sets after deleteing: ",A,",",B)

#Converting the ages to a set and compare the length of the list and the set.
print("\nList of ages: ", age)
set_ages = set(age)
print("\nAges set after converting from list: ", set_ages)
#comparing the length of ages list and ages set
print("\nlength of the age list: ", len(age))
print("\nLength of the age set: ", len(set_ages))
if(len(age)==len(set_ages)):
    print("\nLength of the age list and age set are not equal")
#Length of the set of ages is less compared to length of list of ages as the set contains only unique values
```

```
Length of the it_companies:  7

After adding twitter into the set:  {'Microsoft', 'IBM', 'Facebook', 'Twitter', 'Google', 'Amazon', 'Oracle', 'Apple'}

After adding multiple companies into the set:  {'Microsoft', 'IBM', 'Tesla', 'Google', 'Oracle', 'Twitter', 'Apple', 'Walmart I
nc.', 'Cisco', 'Amazon', 'Samsung', 'Facebook'}

After removing Cisco from the set:  {'Microsoft', 'IBM', 'Tesla', 'Google', 'Oracle', 'Twitter', 'Apple', 'Walmart Inc.', 'Amaz
on', 'Samsung', 'Facebook'}

Set after joining two sets A and B: {19, 20, 22, 24, 25, 26, 27, 28}

Intersection of sets A and B: {19, 20, 22, 24, 25, 26}

Checking if A is subset of B:  True

Checking if A and B are disjoint sets:  False

Joining A with B: {19, 20, 22, 24, 25, 26, 27, 28}

Joining B with A: {19, 20, 22, 24, 25, 26, 27, 28}

Synmmetric difference between A and B:  {27, 28}

A and B sets after deleteing:  set() , set()

List of ages:  [22, 19, 24, 25, 26, 24, 25, 24]

Ages set after converting from list:  {19, 22, 24, 25, 26}

length of the age list:  8

Length of the age set:  5
```

To find the length of the it_companies set I have used len() function.

To add “Twitter” into the set I have used add() method

To insert multiple IT companies to set I have used update() method.

To remove a company from set I have used remove() method.

Here, remove function gives an error if the element is not present in the set, whereas discard function doesn’t give any error if the element is not present in the set.

Next, Join A and B sets I have used union() method.

Then, to find intersection of A and B I have used intersection() method (i.e. A.intersection(B))

To check if A is subset of B, I have used issubset() method, which returns True or False.

Here A = {19, 22, 24, 20, 25, 26} and B = {19, 22, 20, 25, 26, 24, 28, 27} so, we get True.

To check if A and B are disjoint, I have used isdisjoint() method which returns True or False.

Here, in the question as A is subset of B we get False.

In order to join A with B, I have used A.union(B) then, to join B with A, I have used B.union(A)

Next, to obtain the symmetric difference between A and B, I have used symmetric_difference() method.

To delete sets completely, I have used clear() method.

Finally, to convert list to set, I have used set() constructor. Then to check whether their lengths are equal, I have used if else case where in if , I have used == comparator for the length of ages list and length of ages set.

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.

```
import math # Importing math library to get pi value
radius = int(input("Enter radius in meters: ")) # Taking radius as input
_area_of_circle_ = (math.pi)*radius*radius # finding area of the circle using pi*r*r formula
_circum_of_circle_ = 2*(math.pi)*radius # finding circumference of the circle using 2*pi*r formula
print("Area of the circle with radius {} is {}\nCircumference of the circle with radius {} is {}.".format(radius, _area_of_circle
```

```
Enter radius in meters: 30
Area of the circle with radius 30 is 2827.4333882308138
Circumference of the circle with radius 30 is 188.49555921538757.
```

First, I have imported math library in order to get pi value.

I have taken, radius as input from the user, and converting it to int type.

Then calculated area of the circle by using formula math.pi*radius*radius and then the circumference of the circle by using 2*math.pi*radius.

Then I have printed the values using format function in order to replace the {} value with the value in the variable.

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.

```
str1 = "I am a teacher and I love to inspire and teach people"
unique_words_set = set(str1.split(" ")) #converting the string to set by splitting at space
print("Total unique words in the string: ", len(unique_words_set)) #using len() function get total unique words
print("Unique words in the set: ",unique_words_set) # as we are using set, it considers only
```

```
Total unique words in the string:  10
Unique words in the set:  {'I', 'love', 'to', 'teacher', 'am', 'a', 'teach', 'and', 'people', 'inspire'}
```

First, I have initialized string str1 with “I am a teacher and I love to inspire and teach people”.
Then, I in order to get the unique words in the string, I used split(“ ”) function to split the string at space and then converted it to set using set() constructor, as set considers only unique values.
To get the total unique words I have used len() function for the set and then printed the unique words.

Question 7:

Use a tab escape sequence to get the following lines.

Name Age Country City

Asabeneh 250 Finland Helsinki

```
print('Name\tAge\tCountry\tCity\nAsabeneh\t250\tFinland\tHelsinki') # adding a tab space
```

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

I have used "\t" to give tab for each word and "\n" 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."

```
radius = 10
pi = 3.14
area = pi * radius ** 2
result = 'The area of a circle with radius {} is {}  meters square.'.format(radius,area) #using format function to take the value
print(result)
```

```
The area of a circle with radius 10 is 314.0  meters square.
```

I have used format() method to put the values of radius and area in the placeholders of the strings (i.e. {})

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]

```
n = int(input("No.of Students:")) #taking no.of students as input
weight_in_lbs = [] #taking empty list for weight in lbs
print("Enter the weight of students in lbs")
for i in range(n): #looping and storing values of each student weight in the list
    weight_in_lbs.append(int(input()))
weight_in_kgs = [] #taking empty list for weight in kgs
for i in range(n): #looping and converting each weight into kgs and appending into the list
    weight_in_kgs.append(weight_in_lbs[i]* 0.45359237)
print("List of all students weight in kgs: ", weight_in_kgs)
```

```
No.of Students:4
Enter the weight of students in lbs
150
145
155
148
List of all students weight in kgs:  [68.0388555, 65.77089365, 70.30681735, 67.13167076]
```

First, I have taken input from user for the total no.of students and converted it into int.

Then, I have taken an empty list for weight in lbs, which is then appended by taking value from the user using for loop.

Then, I have considered empty list for weight in kgs.

Later, I have converted lbs to kgs for each student by using for loop and multiplying each value with 0.45359237. Then, appended those values to the weight in kgs list

Lastly, I have printed those values.

Question 10:

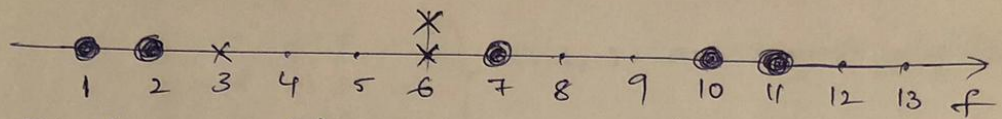
The diagram below shows a dataset with 2 classes and 8 data points, each with only one feature value, labeled f. Note that there are two data points with the same feature value of 6. These are shown as two x's one above the other. Provide stepwise mathematical solution, do not write code for it.

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.

10 Sol)

Given dataset



Dividing the dataset into two equal parts.

let Train dataset = {1, 3, 6, 7}

let test dataset = {2, 6, 10, 11}

KNN Classifier for k = 3

for every testing point we will measure the distance to every training point. to find k = 3 closest points

Step-1 For test data = 2

Find distance from each train point

2 − 1 = 1 ✓
3 − 2 = 1 ✓
6 − 2 = 4 ✓
7 − 2 = 5

} these are first 3 nearest neighbours

Step 2 finding majority class from 3 nearest neighbours

1 ⟶ '●' class, 3 ⟶ 'x' class & 6 ⟶ 'x' class

'x' is majority class

∴ 2 ⟶ belongs to 'x' class

For test data = 6

Step 1: Distance of '6' from each training point

6 − 1 = 5
6 − 3 = 3 ✓
6 − 6 = 0 ✓
6 − 7 = 1 ✓

} 3, 6, 7 are the 3 nearest neighbours

step 2:

3 ⟶ 'x' class, 6 ⟶ 'x' class, 7 ⟶ '●' class.

∴ majority class is 'x' ⇒ ∴ 6 ⟶ belongs to 'x' class

For test data = 10. ; distance from each training point.

$10 - 1 = 9$
$10 - 3 = 7$ ✓
$10 - 6 = 4$ ✓
$10 - 7 = 3$ ✓

} 3, 6, 7 are nearest neighbours

3 belongs to → 'x' class ; 6 → 'x' class and 7 → 'o' class

∴ majority class is 'x' and ; 10 belongs to 'x' class

For test data = 11 ; distance from each training point.

$11 - 1 = 10$
$11 - 3 = 8$ ✓
$11 - 6 = 5$ ✓
$11 - 7 = 4$ ✓

} 3, 6, 7 are nearest neighbours

∴ As 3, 6 → belong to 'x' class which is the majority class.

∴ 11 → belongs to 'x' class

∴ Actual given data set and class

2 → ●
6 → x
10 → ●
11 → ●

Predicted data set and class.

2 → X
6 → X
10 → X
11 → X

Confusion Matrix :

Truth \ Prediction	●	X
●	0 (TN)	3 (FP)
X	0 (FN)	1 (TP)

$$\text{Accuracy} = \frac{TP + TN}{P + N} = \frac{1 + 0}{4} = \frac{1}{4}$$

$$= 0.25$$

Sensitivity or True Positive Rate

$$= \frac{TP}{TP + FN} = \frac{TP}{P} = \frac{1}{1}$$

Specificity or True Negative Rate

$$= \frac{TN}{FP + TN} = \frac{TN}{N} = \frac{0}{}$$