CS5710- Machine Learning

Assignment-1

Name – Sushant Ashish

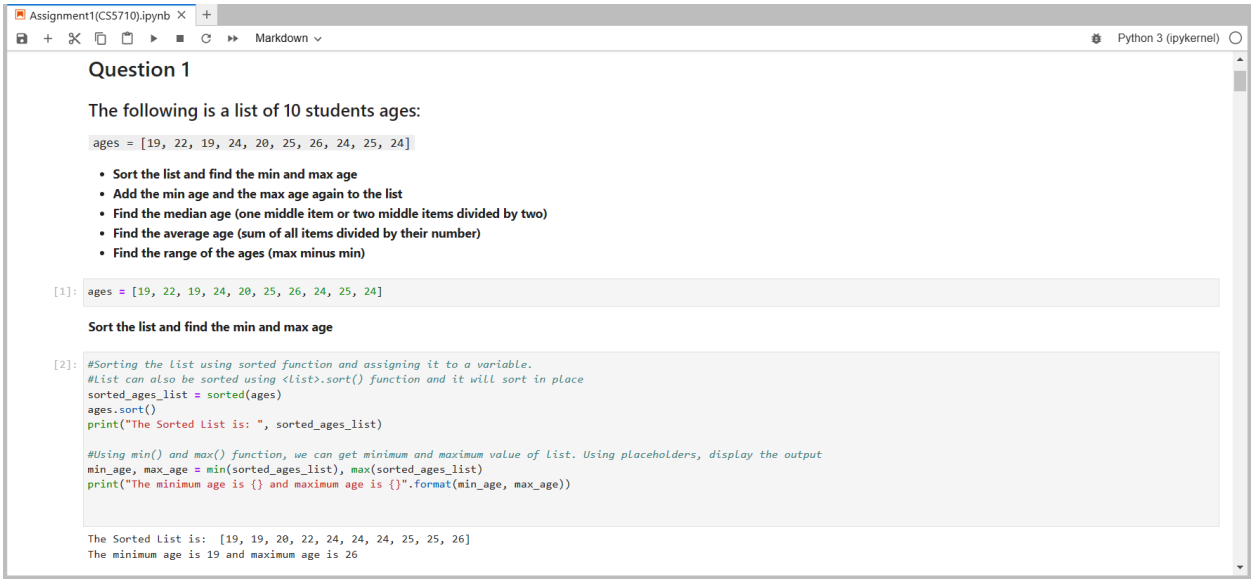
Student ID- 700747009

GitHub Repo Link - https://github.com/sushantUCM/CS5710-Assignment-1.git

Video Demo Link - https://drive.google.com/file/d/19lXDsIgYVSKw5VAc7k1ugquksOdGK2sg/view?usp=share_link

I have created the assignment as a jupyter notebook containing both questions and answers.

Question 10 is also there in the notebook along with my scanned mathematical calculation. The scanned file is also available separately as “Question10-Answer.pdf” in the repository.



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

```
[3]: #Using the extend function we add the min and max age to the list
ages.extend([min_age, max_age])
print(ages)
```

```
[19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 19, 26]
```

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

```
[4]: #Find if the length of list is odd or even
#If length is odd, take center term as median; else, find average of middle 2 terms and serve as median
mid_index = len(ages)//2
if len(ages)%2==0:
    #In this case mid_index = 6, total terms will be 12. Therefore ages[5] and ages[6] average will give correct median
    median = (ages[mid_index] + ages[mid_index - 1]) / 2
else:
    median = ages[mid_index]

print("Median is {}".format(median))
```

```
Median is 24.0
```

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

[5]:
```
#Iterating through the individual items in the list, we can find the total sum of all items and divide by the length of the list
sum = 0
for age in ages:
    sum += age
print("Average age is: {}".format(sum/len(ages)))
```

```
Average age is: 22.75
```

Find the range of the ages (max minus min)

[6]:
```
#We had calculated the min and max age before and later added to the list, making no difference in the resultant min and max value.
print("Range of ages is: {}".format(max_age - min_age))
```

```
Range of ages is: 7
```

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**

Create an empty dictionary called dog

[7]:
```
#Another way to declare empty dictionary is dog = {}
dog = dict()
```

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

[8]:
```
#We will use update() function to update the previously empty dictionary
dog.update({"Name":"Dusty", "Color":"Black", "Breed":"German Shepherd", "Legs":4, "Age":13})
print(dog)
```

```
{'Name': 'Dusty', 'Color': 'Black', 'Breed': 'German Shepherd', 'Legs': 4, 'Age': 13}
```

Create a student dictionary and add first_name, last_name, gender, age, marital status, skills, country, city and address as keys for the dictionary

[9]:
```
#Initializing student dictionary with values
student = {'first_name':'Sushant', 'last_name':'Ashish', 'gender':'Male', 'age':27, 'marital status':'Unmarried', 'skills':['Python','Angular','Git'], 'country':'United States', 'c
print(student)
```

```
{'first_name': 'Sushant', 'last_name': 'Ashish', 'gender': 'Male', 'age': 27, 'marital status': 'Unmarried', 'skills': ['Python', 'Angular', 'Git'], 'country': 'United States', 'ci
ty': 'Kansas City', 'address': 'Holmes Road, Kansas City, MO'}
```

Get the length of the student dictionary

[10]:
```
#Using len function to get length of a dictionary
print("Student dictionary length: ", len(student))
```

```
Student dictionary length:  9
```

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

[11]:
```
#Using type() function to get Data type
print("Data type of skills in student dictionary is {} and values are {}".format(type(student['skills']), student['skills']))
```

```
Data type of skills in student dictionary is <class 'list'> and values are ['Python', 'Angular', 'Git']
```

Modify the skills values by adding one or two skills

[12]:
```
#We will use extend function of list to add 2 skills to skills list in student dictionary
student['skills'].extend(['HTML','Javascript'])
print("Skills of student are: ", student['skills'])
```

```
Skills of student are:  ['Python', 'Angular', 'Git', 'HTML', 'Javascript']
```

Get the dictionary keys as a list

[13]:
```
#We can use casting and keys() function to directly output a list of keys
print("List of keys of student are: ", list(student.keys()))
```

```
List of keys of student are:  ['first_name', 'last_name', 'gender', 'age', 'marital status', 'skills', 'country', 'city', 'address']
```

Get the dictionary values as a list

[14]:
```
#Same as keys, we can use values() function to get values
print("List of values of student are: ", list(student.values()))
```

```
List of values of student are:  ['Sushant', 'Ashish', 'Male', 27, 'Unmarried', ['Python', 'Angular', 'Git', 'HTML', 'Javascript'], 'United States', 'Kansas City', 'Holmes Road, Kansas City, MO']
```

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**

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

[15]:
```
brothers = ('Varun Vishal', 'Aditya Varma')
sisters = ('Surabhi Sinha', 'Anika Kamthan')
print(brothers, sisters)
```

```
('Varun Vishal', 'Aditya Varma') ('Surabhi Sinha', 'Anika Kamthan')
```

Join brothers and sisters tuples and assign it to siblings

[16]:
```
siblings = brothers + sisters
print(siblings)
```

```
('Varun Vishal', 'Aditya Varma', 'Surabhi Sinha', 'Anika Kamthan')
```

How many siblings do you have?

[17]:
```
print("I have {} siblings".format(len(siblings)))
```

```
I have 4 siblings
```

Modify the siblings tuple and add the name of your father and mother and assign it to family_members

[18]:
```
#As tuple is immutable, we cannot modify an existing tuple, therefore creating and assigning family_members
family_members = siblings + ('Ashish Sinha', 'Nirupama Sinha')
print("My family members are: ", family_members)
```

```
My family members are:  ('Varun Vishal', 'Aditya Varma', 'Surabhi Sinha', 'Anika Kamthan', 'Ashish Sinha', 'Nirupama Sinha')
```

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.**

Find the length of the set it_companies

[19]:
```
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 it_companies: ",len(it_companies))
```

```
Length of it_companies:  7
```

Add 'Twitter' to it_companies

[20]:
```
it_companies.add('Twitter')
print(it_companies)
```

```
{'IBM', 'Amazon', 'Oracle', 'Twitter', 'Facebook', 'Apple', 'Microsoft', 'Google'}
```

Insert multiple IT companies at once to the set it_companies

[21]:
```
#add() function adds only 1 entry, for multiple, we use update() and pass a list of items to be added
it_companies.update(['Cisco','Hexagon','Infosys'])
print(it_companies)
```

```
{'IBM', 'Amazon', 'Cisco', 'Hexagon', 'Oracle', 'Twitter', 'Facebook', 'Apple', 'Microsoft', 'Infosys', 'Google'}
```

Remove one of the companies from the set it_companies

[22]:
```
#Using discard() function to remove a specific item given as parameter, if the first element is to be removed, then we can use pop()
it_companies.discard('Oracle')
print(it_companies)
```

```
{'IBM', 'Amazon', 'Cisco', 'Hexagon', 'Twitter', 'Facebook', 'Apple', 'Microsoft', 'Infosys', 'Google'}
```

What is the difference between remove and discard

[23]:
```
#discard() function performs a check if the intended item to remove exists previously or not. remove() should throw an error

#remove() - removing non-existent element
it_companies.remove('Oracle')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_32156\7430138.py in <module>
      2
      3 #remove() - removing non-existent element
----> 4 it_companies.remove('Oracle')

KeyError: 'Oracle'
```

[24]:
```
# discard() - trying to discard same non-existent element
it_companies.discard('Oracle')
print(it_companies)
```

```
{'IBM', 'Amazon', 'Cisco', 'Hexagon', 'Twitter', 'Facebook', 'Apple', 'Microsoft', 'Infosys', 'Google'}
```

Join A and B

[25]:
```
#Joining 2 sets i.e. finding union of 2 sets
join_A_B = A.union(B)
print("Union of both sets: ", join_A_B)
```

```
Union of both sets:  {19, 20, 22, 24, 25, 26, 27, 28}
```

Find A intersection B

[26]:
```
#Using intersection() function
intersection_A_B = A.intersection(B)
print("A intersection B is: ", intersection_A_B)
```

```
A intersection B is:  {19, 20, 22, 24, 25, 26}
```

Is A subset of B

[27]:
```
#Using issubset() function
A_subset_of_B = A.issubset(B)
print("Is A subset of B: ", A_subset_of_B)
```

```
Is A subset of B:  True
```

Are A and B disjoint sets

[28]:
```
#isdisjoint() compares both sets together, so precendence doesn't matter
print("Are A and B disjoint sets: ", A.isdisjoint(B))
```

```
Are A and B disjoint sets:  False
```

Join A with B and B with A

[29]:
```
#Using update() function, the set on which operation is performed is the base, and set in parameter is the one which will be added into the base
#Adding A to B
B.update(A)
print("Join A with B: ", B)

#Adding B to A
B_to_A = A.update(B)
print("Join B with A: ", A)

#Checking if both results are same
if A == B:
    print("Both sets are same")
```

```
Join A with B:  {19, 20, 22, 24, 25, 26, 27, 28}
Join B with A:  {19, 20, 22, 24, 25, 26, 27, 28}
Both sets are same
```

What is the symmetric difference between A and B

[30]:
```
#Using symmetric_difference() or symmetric_difference_update()
sym_diff = A.symmetric_difference(B)
print("Symmetric difference is: ", sym_diff)
```

```
Symmetric difference is:  set()
```

Delete the sets completely

[31]:
```
#Using clear() function
A.clear()
B.clear()

print(A, B)
```

```
set() set()
```

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

[32]:
```
#By casting we can convert a list to set
age_set = set(age)
print("List: ", age)
print("Set: ", age_set)

#Set removes duplicate items
print("Length of list is {} and that of set is {}".format(len(age), len(age_set)))
```

```
List:  [22, 19, 24, 25, 26, 24, 25, 24]
Set:  {19, 22, 24, 25, 26}
Length of list is 8 and that of set is 5
```

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.**

Calculate the area of a circle and assign the value to a variable name of `_area_of_circle_`

[33]:
```
#Assigning radius and value of pi, area= pi*r*r
#Assuming pi=3.14

pi = 3.14
radius = 30
_area_of_circle_ = pi*(radius**2)
print("Area of circle: {} m*m".format(_area_of_circle_))

#We can get more accurate value if we use Math module
import math
_area_of_circle_ = math.pi*(radius**2)
print("Area of circle(Math module): {} m*m".format(_area_of_circle_))
```

```
Area of circle: 2826.0 m*m
Area of circle(Math module): 2827.4333882308138 m*m
```

Calculate the circumference of a circle and assign the value to a variable name of `_circum_of_circle_`

[34]:
```
#Circumference = 2*pi*r

_circum_of_circle = 2*pi*radius
print("Circumference of circle is: ", _circum_of_circle)
```

```
Circumference of circle is:  188.4
```

Take radius as user input and calculate the area.

[35]:
```
#Using input() or raw_input() to take input from user and parsing input as float or int
radius = float(input("Enter the radius of circle: "))
_area_of_circle_ = pi*(radius**2)
print("Area: ", _area_of_circle_)
```

```
Enter the radius of circle:  14
Area:  615.44
```

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**

[36]:
```
#Using split function of string, we will split words with spaces into a list and convert into a set, so that it removes duplicate values
sentence = 'I am a teacher and I love to inspire and teach people'
list_words = sentence.split(' ')
set_words = set(list_words)
print("The number of unique words is {} which are as follows: {}".format(len(set_words), set_words))
```

```
The number of unique words is 10 which are as follows: {'a', 'teach', 'and', 'I', 'love', 'to', 'am', 'people', 'inspire', 'teacher'}
```

Question 7

Use a tab escape sequence to get the following lines.

Name Age Country City
Asabeneh 250 Finland Helsinki

[37]:
```
#Using \t and \n escape characters we can format
print("Name\tAge\tCountry\tCity\nAsabeneh\t250\tFinland\tHelsinki")
```

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

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."`

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]`

[39]:
```
number_of_students = int(input("Enter number of Students: "))
input_weights = []
output_weights = []
for i in range(number_of_students):
    weight_pounds = float(input("Weight in pounds: "))
    input_weights.append(weight_pounds)
    #1lb = 0.45359237kg
    weight_kg = round(weight_pounds * 0.45359237, 2)
    output_weights.append(weight_kg)

print("L1: ", input_weights)
print("Output: ", output_weights)
```

```
Enter number of Students:  5
Weight in pounds:  150
Weight in pounds:  155
Weight in pounds:  145
Weight in pounds:  148
Weight in pounds:  200
L1:  [150.0, 155.0, 145.0, 148.0, 200.0]
Output:  [68.04, 70.31, 65.77, 67.13, 90.72]
```