MACHINE LEARNING

ASSIGNMENT – 1

Chenna Praneeth Reddy Kesara

700756460

GIT HUB: https://github.com/reddy8888/ML_Assignment1

```
# Given list of ages
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

# Sort the list and find the min and max age
ages.sort()
min_age = ages[0]
max_age = ages[-1]

# Add the min age and the max age again to the list
ages.append(min_age)
ages.append(max_age)
ages.sort()  # Sort again after adding

# Find the median age
n = len(ages)
if n % 2 == 0:
    median = (ages[n//2 - 1] + ages[n//2]) / 2
else:
    median = ages[n//2]

# Find the average age
average_age = sum(ages) / len(ages)

# Find the range of the ages
age_range = max_age - min_age
```

```
# Output the results
print(f"Sorted list: {ages}")
print(f"Min age: {min_age}")
print(f"Max age: {max_age}")
print(f"Median age: {median}")
print(f"Average age: {average_age}")
print(f"Range of ages: {age_range}")
```

```
Sorted list: [19, 19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 26]
Min age: 19
Max age: 26
Median age: 24.0
Average age: 22.75
Range of ages: 7
```

Question 2

```python
# Create an empty dictionary called dog
dog = {}

# Add name, color, breed, legs, age to the dog dictionary
dog['name'] = 'Buddy'
dog['color'] = 'Brown'
dog['breed'] = 'Golden Retriever'
dog['legs'] = 4
dog['age'] = 3

# Create a student dictionary and add various keys
student = {
    'first_name': 'John',
    'last_name': 'Doe',
    'gender': 'Male',
    'age': 20,
    'marital_status': 'Single',
    'skills': ['Python', 'Data Analysis'],
    'country': 'USA',
    'city': 'New York',
    'address': '1234 Elm Street'
}

# Get the length of the student dictionary
student_length = len(student)

# Get the value of skills and check the data type
skills = student['skills']
skills_type = type(skills)
```

```python
# Get the value of skills and check the data type
skills = student['skills']
skills_type = type(skills)

# Modify the skills values by adding one or two skills
student['skills'].extend(['Machine Learning', 'Communication'])

# Get the dictionary keys as a list
student_keys = list(student.keys())

# Get the dictionary values as a list
student_values = list(student.values())

# Output the results
print(f"Dog dictionary: {dog}")
print(f"Student dictionary: {student}")
print(f"Length of student dictionary: {student_length}")
print(f"Skills: {skills}")
print(f"Type of skills: {skills_type}")
print(f"Student dictionary keys: {student_keys}")
print(f"Student dictionary values: {student_values}")
```

```
Dog dictionary: {'name': 'Buddy', 'color': 'Brown', 'breed': 'Golden Retriever', 'legs': 4, 'age': 3}
Student dictionary: {'first_name': 'John', 'last_name': 'Doe', 'gender': 'Male', 'age': 20, 'marital_status': 'Single', 'skills': ['Python', 'Data Analysis', 'Machine Learning', 'Comm
Length of student dictionary: 9
Skills: ['Python', 'Data Analysis', 'Machine Learning', 'Communication']
Type of skills: <class 'list'>
Student dictionary keys: ['first_name', 'last_name', 'gender', 'age', 'marital_status', 'skills', 'country', 'city', 'address']
Student dictionary values: ['John', 'Doe', 'Male', 20, 'Single', ['Python', 'Data Analysis', 'Machine Learning', 'Communication'], 'USA', 'New York', '1234 Elm Street']
```

Question 3

```python
# Create a tuple containing names of your sisters and brothers (imaginary siblings are fine)
sisters = ('Anna', 'Maria')
brothers = ('John', 'Alex')

# Join brothers and sisters tuples and assign it to siblings
siblings = sisters + brothers

# How many siblings do you have?
num_siblings = len(siblings)

# Modify the siblings tuple by converting it to a list (since tuples are immutable),
# making the modification, and then converting it back to a tuple
siblings_list = list(siblings)
siblings_list.append('NewSibling')  # Adding a new sibling as an example
siblings = tuple(siblings_list)

# Output the results
print(f"Sisters: {sisters}")
print(f"Brothers: {brothers}")
print(f"Siblings: {siblings}")
print(f"Number of siblings: {num_siblings}")
```

```
Sisters: ('Anna', 'Maria')
Brothers: ('John', 'Alex')
Siblings: ('Anna', 'Maria', 'John', 'Alex', 'NewSibling')
Number of siblings: 4
```

Question 4

```python
# Given sets and list
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
length_it_companies = len(it_companies)
print(f"Length of it_companies: {length_it_companies}")

# Add 'Twitter' to it_companies
it_companies.add('Twitter')
print(f"it_companies after adding 'Twitter': {it_companies}")

# Insert multiple IT companies at once to the set it_companies
it_companies.update(['Spotify', 'Netflix', 'Adobe'])
print(f"it_companies after adding multiple companies: {it_companies}")

# Remove one of the companies from the set it_companies
it_companies.remove('IBM')  # Remove 'IBM' as an example
print(f"it_companies after removing 'IBM': {it_companies}")

# Difference between remove and discard
# remove() will raise a KeyError if the element does not exist
# discard() will not raise an error if the element does not exist

# Join A and B
A_union_B = A.union(B)
print(f"Union of A and B: {A_union_B}")
```

```python
# Find A intersection B
A_intersection_B = A.intersection(B)
print(f"Intersection of A and B: {A_intersection_B}")

# Is A subset of B
is_A_subset_of_B = A.issubset(B)
print(f"Is A a subset of B: {is_A_subset_of_B}")

# Are A and B disjoint sets
are_A_and_B_disjoint = A.isdisjoint(B)
print(f"Are A and B disjoint sets: {are_A_and_B_disjoint}")

# Join A with B and B with A (Union of A and B is commutative)
A_union_B = A.union(B)
B_union_A = B.union(A)
print(f"A union B: {A_union_B}")
print(f"B union A: {B_union_A}")

# What is the symmetric difference between A and B
A_symmetric_difference_B = A.symmetric_difference(B)
print(f"Symmetric difference between A and B: {A_symmetric_difference_B}")

# Delete the sets completely
del A
del B

# Convert the ages list to a set and compare the length of the list and the set
age_set = set(age)
length_age_list = len(age)
length_age_set = len(age_set)
print(f"Length of age list: {length_age_list}")
```

```python
# What is the symmetric difference between A and B
A_symmetric_difference_B = A.symmetric_difference(B)
print(f"Symmetric difference between A and B: {A_symmetric_difference_B}")

# Delete the sets completely
del A
del B

# Convert the ages list to a set and compare the length of the list and the set
age_set = set(age)
length_age_list = len(age)
length_age_set = len(age_set)
print(f"Length of age list: {length_age_list}")
print(f"Length of age set: {length_age_set}")
```

```
Length of it_companies: 7
it_companies after adding 'Twitter': {'Microsoft', 'Apple', 'Google', 'IBM', 'Oracle', 'Amazon', 'Facebook', 'Twitter'}
it_companies after adding multiple companies: {'Microsoft', 'IBM', 'Netflix', 'Amazon', 'Twitter', 'Apple', 'Facebook', 'Oracle', 'Google', 'Adobe', 'Spotify'}
it_companies after removing 'IBM': {'Microsoft', 'Netflix', 'Amazon', 'Twitter', 'Apple', 'Facebook', 'Oracle', 'Google', 'Adobe', 'Spotify'}
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 a subset of B: True
Are A and B disjoint sets: False
A union B: {19, 20, 22, 24, 25, 26, 27, 28}
B union A: {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric difference between A and B: {27, 28}
Length of age list: 8
Length of age set: 5
```

Question 5

```python
import math

# Given radius
radius = 30

# Calculate the area of the circle
_area_of_circle_ = math.pi * (radius ** 2)
print(f"Area of the circle with radius {radius} meters: {_area_of_circle_:.2f} square meters")

# Calculate the circumference of the circle
_circum_of_circle_ = 2 * math.pi * radius
print(f"Circumference of the circle with radius {radius} meters: {_circum_of_circle_:.2f} meters")

# Take radius as user input and calculate the area
user_radius = float(input("Enter the radius of the circle in meters: "))
user_area_of_circle = math.pi * (user_radius ** 2)
print(f"Area of the circle with radius {user_radius} meters: {user_area_of_circle:.2f} square meters")
```

```
Area of the circle with radius 30 meters: 2827.43 square meters
Circumference of the circle with radius 30 meters: 188.50 meters
Enter the radius of the circle in meters: 30
Area of the circle with radius 30.0 meters: 2827.43 square meters
```

Question 6

```python
# Given sentence
sentence = "I am a teacher and I love to inspire and teach people"

# Split the sentence into words
words = sentence.split()

# Convert the list of words to a set to get unique words
unique_words = set(words)

# Find the number of unique words
num_unique_words = len(unique_words)

# Output the result
print(f"Number of unique words: {num_unique_words}")
print(f"Unique words: {unique_words}")
```

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

Question 7

```
# Use tab escape sequences to format the lines
print("Name\t\tAge\tCountry\tCity")
print("Asabeneh\t250\tFinland\tHelsinki")
```

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

Question 8

```
# Read the number of students (N) from the user
N = int(input("Enter the number of students: "))

# Initialize an empty list to store weights in pounds
weights_lbs = []

# Read weights of N students into the list
for i in range(N):
    weight = float(input(f"Enter weight of student {i+1} in pounds: "))
    weights_lbs.append(weight)

# Convert weights from pounds to kilograms and store them in a separate list
weights_kgs = []
for weight_lbs in weights_lbs:
    weight_kg = weight_lbs * 0.453592  # Conversion factor: 1 pound = 0.453592 kilograms
    weights_kgs.append(round(weight_kg, 2))  # Round to two decimal places

# Output the weights in kilograms
print("Weights in kilograms:")
print(weights_kgs)S
```

```
Enter the number of students: 4
Enter weight of student 1 in pounds: 120
Enter weight of student 2 in pounds: 130
Enter weight of student 3 in pounds: 140
Enter weight of student 4 in pounds: 150
Weights in kilograms:
[54.43, 58.97, 63.5, 68.04]
```