**Name : Bonu Manoj Kumar**

**ID : 700740713**

**Git Link:** [mxb07130/MachineLearning\_Assignments: Machine Learning Assignments (github.com)](https://github.com/mxb07130/MachineLearning_Assignments)

**Machine Learning Assignment 1**

**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)

**Program:**

import statistics

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

#sorting the list and showing min and max values

sortedlist = ages.sort()

print(ages)

print(min(ages))

print(max(ages))

#adding min and max values to list

minvalue=min(ages)

maxvalue=max(ages)

ages.append(minvalue)

ages.append(maxvalue)

print(ages)

#Finding median by importing statistics package

median=statistics.median(ages)

print(median)

#Finding average using for loop

length=len(ages)

a=0

for i in ages:

    a+=i

print(a/length)

#Range value

ranges=maxvalue-minvalue

print(ranges

**Output:**

Text

Description automatically generated

**Explantion:**

* **Sorting the list using SORT() function and printing the updated list ages**
* **Finding the min and max value from the list ages using MIN(),MAX() functions and storing it in minimum and maximum and then printing both the values**
* **Adding the minimum and maximum to the list ages again using the append function**
* **Finding the median sum length average using the math functions**
* **Calculated the range by finding the difference between maximum and minimum values from list and sorting and printing the range**

**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, 1ast\_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

**Program :**

dog={}

#updated dog dictionary

dog={"name":"cherry","color":"red","breed":"pug","legs":"short","age":5}

#creating student dictionary

student={"first\_name":"manoj","last\_name":"kumar","gender":"male","age":23,"martial status":"single","skills":["java","c","python"],"country":"usa","city":"kansas","address":"1004"}

#length

print("Length:",len(student))

#values of skills

print("Skills:",student["skills"])

#modify

student["skills"].append("c sharp")

print("Modified Skills:",student["skills"])

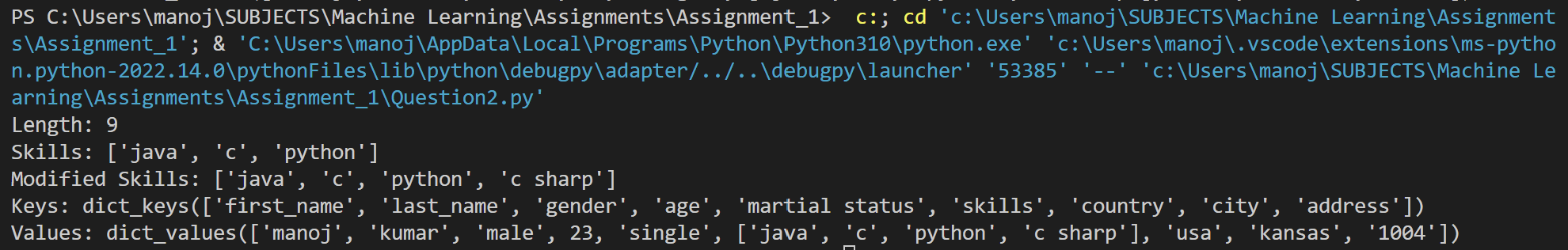
#printing keys

print("Keys:",student.keys())

#printing values

print("Values:",student.values())

**Output:**

****

**Explanation:**

* **Creating an empty dictionary dog and updating the values in the dictionary**
* **Calculating the length of the student dictionary using len function**
* **Modifying the skills value by using the extend() function**
* **Printing the keys in the student dictionary using the functions 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 fami1y\_member

**Program:**

# tuple with names of sisters and brothers

sisters = ("Kavya", "Ramya", "lavanya", "Sirisha", "Sravanthi")

brothers = ("Manoj", "Narendra", "Sravan")

print(sisters)

print(brothers)

# joining brothers and sisters

siblings= sisters + brothers

print(siblings)

#count of siblings

count\_siblings= len(siblings)

print(count\_siblings)

#appending father and mother

mother = "Latha"

father= "Krishna\_Mohan"

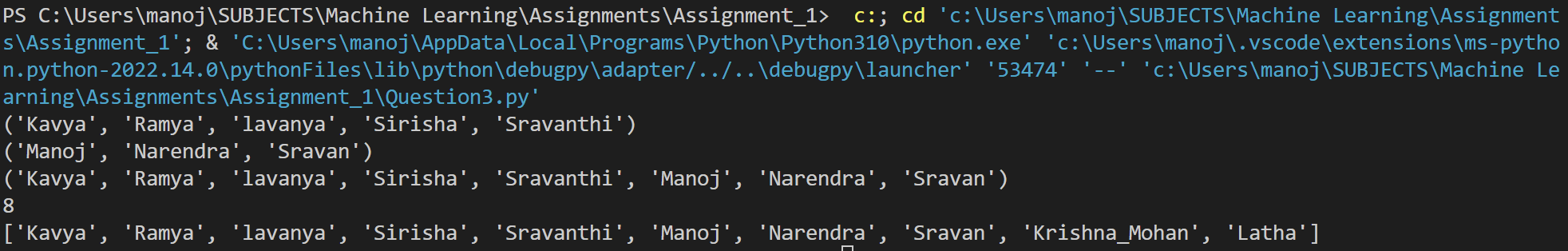
family\_members = list(siblings)

family\_members.append(father)

family\_members.append(mother)

print(family\_members)

**Output:**



**Explanation:**

* **Create tuples with values assigned**
* **Creating a tuple for brothers and sisters**
* **Joining both the tuples and result it as siblings tuple**
* **Count the number of siblings using len() function that gives the length of tuple**
* **Assign the values of mother and father**
* **Append the father and mother list to the family members list using append()**
* **Convert the list intlo 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.

**Program:**

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

print("The length of set is:", len(it\_companies))

#Add 'Twitter' to it\_companies

it\_companies.add("Twitter")

print(it\_companies)

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

multiple\_ITcompanies= ["Tesla","Samsung", "Deloitte", "Meta"]

it\_companies.update(multiple\_ITcompanies)

print(it\_companies)

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

it\_companies.remove("Samsung")

print(it\_companies)

#Join A and B

C = A.union(B)

print(C)

#Find A intersection B

D = A.intersection(B)

print(D)

#Is A subset of B

E = A.issubset(B)

print(E)

#Are A and B disjoint sets

F = A.isdisjoint(B)

print(F)

#Join A with B and B with A

G = B.union(A)

print(C,G)

#What is the symmetric difference between A and B

H = A.symmetric\_difference(B)

print(H)

#Delete the sets completely

del A,B

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

I= set(age)

print(len(I))

**Output:**

Text

Description automatically generated with medium confidence

**Explanation:**

* **Find the length of the it\_companies list using len() function and printing the result**
* **Modifying an elemnt in the it\_companies using append()**
* **Inserting multiple companies using the update() function**
* **Removing an IT company from the list using the remove() function**
* **Joining A and B list using the union() function**
* **Intersection of A list with B and printing the intersection list**
* **Found if A is a subset of B using the is.subset() function**
* **Foind if A and B are disjoint sets using is.disjoint() function**
* **Found the symmetric\_difference between A and B using the symmetric\_difference() function**
* **Deleting the sets completely**
* **Converting the agest into a set using set() function and computing the length of the list and set using len function which gives a result**

**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\_circ1e\_

* Take radius as user input and calculate the area.

**Program:**

radius = 30

pi=3.14

#area of a circle

area\_of\_circle = pi\*radius\*radius

print("area of the circle is", area\_of\_circle)

#circumference of a circle

circum\_of\_cirlce = 2\*pi\*radius

print("circumference of the circle is", circum\_of\_cirlce)

#calculating area with radius as input

radius = float(input ("Enter the radius of the circle : "))

area= pi\*radius\*radius

print ("The area of the circle is", area)

**Output:**

Text

Description automatically generated

**Explanation:**

* **Calculated the area and circumference of the circle using the formula and randomly taking an input value for the radius and took an input for radius and calculated the area of the circle**

**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.

**Program:**

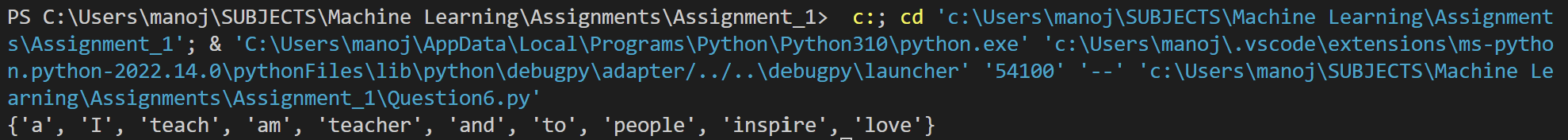
#printing unique words

input1 =  "I am a teacher and I love to inspire and teach people"

strsplit=set(input1.split(" "))

print(strsplit)

**Output:**

****

**Explanation:**

* **Printing the unique words from the given sentence using set and split function where split function splits the string into the unique words**

**Question 7:**

Use a tab escape sequence to get the following lines.

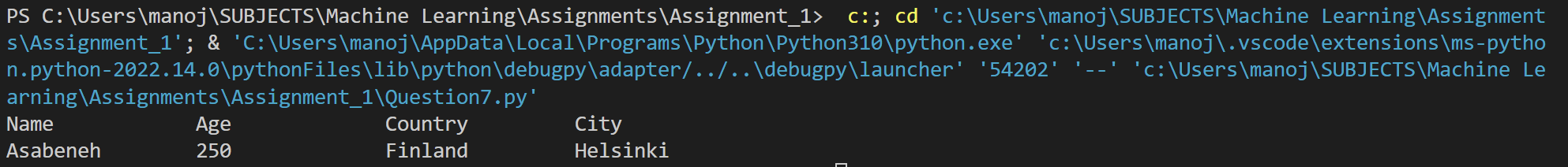
# Name Age Country City Asabeneh 250 Finland Helsinki

**Program:**

#printing the sequence using escape tab

print(“Name\t\tAge\t\tCountry\t\tCity\nAsabeneh\t250\t\tFinland\t\tHelsinki”)

**Output:**



**Explanation:**

* **Created a pattern using \n which is for newline and \t for tabspace**

**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.”

**Program:**

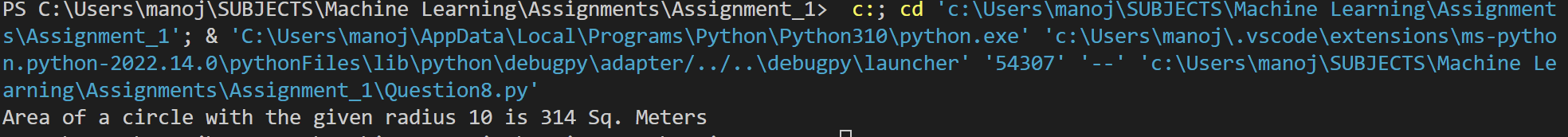
radius = 10

area = 3.14 \* radius \*\* 2

# printing by string formatting method

print("Area of a circle with the given radius {} is {} Sq. Meters".format(radius,int(area)))

**Output:**



**Explanation:**

* **Used the string formatting methods to calculate the area of the circle**

**Question 9:**

Write a program, which reads weights (lbs.) ofN 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]

**Program:**

import math

# Taking the integer value as input

students = int(input("Enter the count of all the students: "))

list=[]

list1=[]

# Taking all inputs into list

for i in range(students):

   list.append(int(input()))

# coverting lbs into kgs by using math.floor

for m in list:

    x=(math.floor((m/2.2046) \* 100 ))/ 100;

    list1.append(x)

print("Weights in kg is:", list1)

**Output:**

Text, application

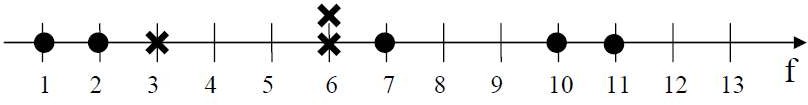
Description automatically generated

**Explanation:**

* **Converted from lbs to kgs we are giving a formula where lbs is divided by the value 2.2046 and multiply and divide with 100 for number formatting and applying them to all the list 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.



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.

**Program:**

import numpy as np #importing important python libraries  
import matplotlib.pyplot as plt   
import pandas as pd  
path=r"C:\Users\manoj\SUBJECTS\Machine Learning\Assignments\Assignment\_1\dataset .csv"  
data=pd.read\_csv(path)#reading the dataset  
print(data)  
ab= data['Feature'].values  
bc= data['Class'].values  
print(ab,bc)  
#dividing data equally into training and testing data  
from sklearn.model\_selection import train\_test\_split   
features\_tr, features\_te, label\_tr, label\_te= train\_test\_split(ab, bc, random\_state=0, train\_size= 0.5 )  
#reshaping the data feature and labes into 2D array  
features\_tr = np.array(features\_tr).reshape(-1,1)  
features\_te = np.array(features\_te).reshape(-1,1)  
#Normalizing data  
from sklearn.preprocessing import StandardScaler   
normalization= StandardScaler()   
features\_tr= normalization.fit\_transform(features\_tr)   
features\_te= normalization.transform(features\_te)   
#fiting the training data into classifier model   
from sklearn.neighbors import KNeighborsClassifier   
model= KNeighborsClassifier(n\_neighbors=3 )   
model.fit(features\_tr, label\_tr)  
#Predicting the test set result   
predict\_class= model.predict(features\_te)   
print("Predicted Test Samples Output:",predict\_class)  
  
#creating a confusion matrix  
from sklearn.metrics import confusion\_matrix   
model\_evaluation= confusion\_matrix(label\_te, predict\_class)   
print("Confusion matrix:\n",model\_evaluation)  
#finding model accuracy  
count=sum(sum(model\_evaluation))  
accuracy=(model\_evaluation[0,0]+model\_evaluation[1,1])/count  
print ('Accuracy =: ', accuracy)  
# finding model sensitivity  
sense = model\_evaluation[0,0]/(model\_evaluation[0,0]+model\_evaluation[0,1])  
print('Sensitivity =: ', sense )  
#finding model specificity  
speci = model\_evaluation[1,1]/(model\_evaluation[1,0]+model\_evaluation[1,1])  
print('Specificity =: ', speci)

**Output:**

Text

Description automatically generated

**Explanation:**

* **Importing the libraries numpy and matplotlib**
* **Reading the dataset by giving the path**
* **Divide dataset into train and test**
* **Fit the training dataset into the model classifier**
* **And then predict the test set result**
* **Create a model accuracy and find it**
* **Find the sensitivity and specificity**