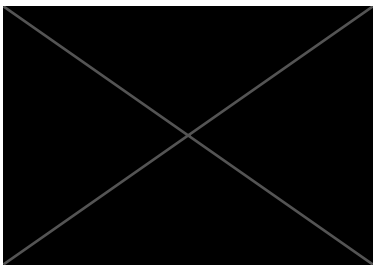
Deakin University

Machine Learning

OnTrack Submission

Task 1.1P



Tutor:
Xinzhe Li

March 5, 2024

[redacted]

1.1. Summarise the main points that is covered in this week.

This being the first week of [redacted]

1.2. Provide summary of your reading list – external resources, websites, book chapters, code libraries, etc.

My reading list for this week [redacted]

1.3. Reflect on the knowledge that you have gained by reading contents of this week with respect to machine learning.

As touched on above, a lot of the knowledge gained this week is [redacted]

1.4. Attempt the quiz given in weekly content (1.28) and add screenshot of your score (>85% is considered completion of this task) in this report.

Your work has been saved and submitted

[redacted]

Your quiz has been submitted successfully, the answer(s) for the following question(s) are incorrect.

Attempt Score 10 / 10 - 100 %

Overall Grade (Highest Attempt) 10 / 10 - 100 %

Quiz List › Submissions

Quiz Submissions - Week-1 quiz

Add to ePortfolio

[redacted]

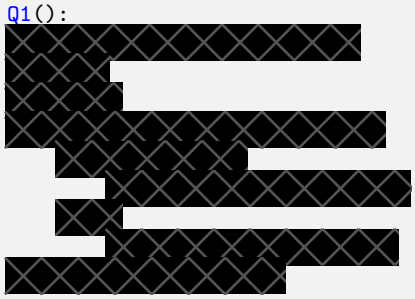
Individual Attempts	Grade
Attempt 1	8 / 10 - 80 %
Attempt 2	10 / 10 - 100 %
Overall Grade (highest attempt):	10 / 10 - 100 %

Cell 01

```
import numpy as np
```

Cell 02

```
#Define a string variable with your name. Print characters of odd and even position of your
↪  name.
def Q1():
```

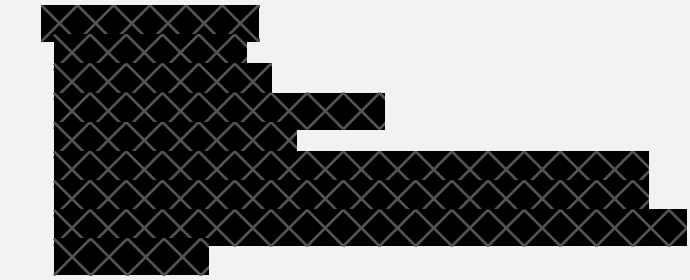


```
Q1()
```

```
['                    's']
 ['W',                   ]
```

Cell 03

```
#Create three variables named 'Student name', 'Course name', and 'EmailAddress'. Add 5 random
↪  values in these variables, where added values are not replaceable and print them.
from string import ascii_letters
from random import shuffle
def Q2():
```



```
Q2()
```

```
('fidjt',      , 'okgrs')
```

Cell 04

```
#Create a dictionary variable with three keys, which are named same as the three variables
↪  created in the previous problem. Insert the previously created three variables values in
↪  the dictionary variable and print it.
def Q3():
```

```
Q3()



{'studnetName': 'sorje', 'courseName': 'abdmi', 'emailAddress': 'ktglp'}
```

Cell 05

```
#Define a string variable 'energy' and assign value either 'Fossil' or 'Renewable'. Print
↪  "Natural gas" if the value of energy variable is 'Fossil' otherwise print "solar power".
↪  What will your program print if energy='Biomass '?
def Q4():
    energy = 'Fossil'
    if energy.lower() == 'fossil':
        print("Natural Gas")
    else:
        print("solar power")
    print("If energy is equal to 'Biomass' then 'solar power' will be printed as as control
    ↪  flow default to the else condition if the string does not match 'fossil'")
Q4()
```

```
Natural Gas
If energy is equal to 'Biomass' then 'solar power' will be printed as as control flow default
↪  to the else condition if the string does not match 'fossil'
```

Cell 06

```
#Print all prime numbers between 0 and 100 separated by line.
def Q5():
    [redacted]
        print(temp)
```

Cell 07

```
[redacted]
Q5()
```

```
Testing function against list of known primes in range......function matches test case:
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89,
↪  97]
```

Cell 08

```
#Is it possible to break a loop before executing for the defined number of iterations? If yes,
↪  provide an example using While loop structure. Otherwise, explain your answer (why and
↪  how).
def Q6():
    [redacted]
Q6()
```

```
print numbers from 0 to 100:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
↪  35 36 37 38 39 40 41 42 exiting loop early...
```

Cell 09

```
#Create  functions named  'Addition', 'Multiplication', 'Division' and 'Subtraction'. These
↪  functions takes two input parameters say (x, y) and return (x+y),(x*y), (x/y) and (x-y).
↪  Call the function with (4,7) arguments and print the outputs
def Q7():
    x = 4
    y = 7
     rint("Add:", Addition(x, ))
[redacted]

def Addition(x,y):
    return x + y
def Multi lication(x,y):
    [redacted]
def Division(x, ):
    [redacted]
def Subtraction(x,y):
    [redacted]

Q7()
```

```
Add: 11
Mul: 28
Div: 0.5714285714285714
Sub: -3
```

Cell 10

```
#Create and display a 8x8 Identity matrix and multiply it with a 8x4 Random matrix.
def Q8():
    idMatrix = n .identity(8)
    [redacted]
    print(np.matmul(idMatrix, rndMatrix))
Q8()
```

```
----------------------------------------------------------------------------------------------------


 [[1. 0. 0. 0. 0. 0. 0.]
  [0. 1. 0. 0. 0. 0. 0.]
  [0. 0. 1. 0. 0. 0. 0.]
  [0. 0. 0. 1. 0. 0. 0.]
  [0. 0. 0. 0. 1. 0. 0.]
  [0. 0. 0. 0. 0. 1. 0.]
  [0. 0. 0. 0. 0. 0. 1.]]
 [[5 1 5 3]
  [3 5 5 6]
  [5 8 5 5]
  [3 1 6 1]
  [5 8 5 3]
  [5 1 2 8]
  [6 0 7 7]
  [2 1 0 3]]
 [[5. 1. 5. 3.]
  [3. 5. 5. 6.]
  [5. 8. 5. 5.]
  [3. 1. 6. 1.]
  [5. 8. 5. 3.]
  [5. 1. 2. 8.]
  [6. 0. 7. 7.]
  [2. 1. 0. 3.]]
```

Cell 11

```
#Create a 8x9 matrix, initialise with random values and print it. Now, transpose the matrix and print it. How the transposed matrix is different from original matrix?
def Q9():
    rndMatrix = n .random.randint(9,size=(8,9))
    [redacted]
    print("The transposed matrix is a diagonal flip of the original matrix. If the matrix was printed on a transparent piece of paper this would have")
    print("the effect of picking the paper up, flipping it over, and rotating it 90 degrees counter clockwise")
Q9()
```

```
[[5 5 3 2 4 3 3 7 5]
 [4 3 8 8 6 7 2 2 3]
 [1 7 2 1 7 6 6 4 3]
 [1 7 2 5 3 7 6 3 4]
 [8 5 8 0 0 8 1 8 0]
 [6 5 3 2 4 6 2 3 5]
 [8 6 6 8 6 6 4 6 0]
 [5 6 7 6 6 0 1 7 7]]
[[5 4 1 1 8 6 8 5]
 [5 3 7 7 5 5 6 6]
 [3 8 2 2 8 3 6 7]
 [2 8 1 5 0 2 8 6]
 [4 6 7 3 0 4 6 6]
 [3 7 6 7 8 6 6 0]
 [3 2 6 6 1 2 4 1]
 [7 2 4 3 8 3 6 7]
 [5 3 3 4 0 5 0 7]]
The transposed matrix is a diagonal flip of the original matrix. If the matrix was printed on a transparent piece of paper this would have
the effect of picking the paper up, flipping it over, and rotating it 90 degrees counter clockwise
```

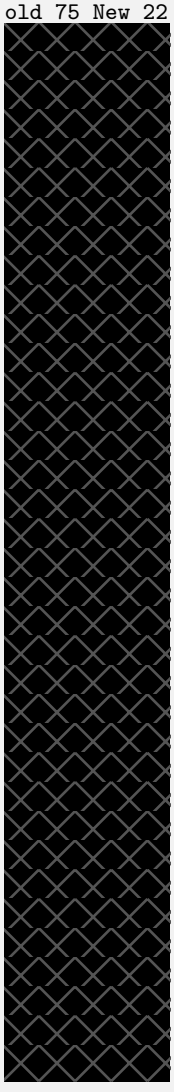
Cell 12

```
#Expport the "PedestrianCounting.csv" file as a matrix. The row of the matrix represents data point or instance and the column represents the variable or feature.
#Print the number of data instances and variables present in this dataset. Calculate mean of the third variable and deduct this mean from original values of the variable.
#Print both original and modified values of the variable.
```

[redacted]

```
def Q10():
    tem  =
    print("Number of data instances (rows):", numRows)
    #number of values is the number of rows multiplied by the number of columns
    cols = len(temp[0])
    print("Number of variables (cols):     ", cols)
    print("Total number of variables:      ", cols * numRows) #Aassuming there are no blanks
    np.transpose(temp)
    print("Average location ID:", GetMean(temp))
    print("Applying mean to third value of matrix")
    ApplyMean(temp, GetMean(temp))
Q10()
```

```
Number of data instances (rows): 515361
Number of variables (cols):      8
Total number of variables:       4122888
Average location ID: 53
Applying mean to third value of matrix
old 75 New 22
```



[redacted]