MACHINE LEARNING ASSIGNMENT 2

K RAHUL RAO

Amrita Vishwa Vidyapeetham

22AIE213 MACHINE LEARNING

Functions used in Rahul’s questions

Question 1: finding euclidean’s distance

We create function called Euclidean diatance , iterate through its range and we take dimension

Values. After getting the dimensions we use the f-string to format later we append the values and

Return the function . the vectors and its dimensions are taken from the user input

Pseudo code =

from math import sqrt

function Create\_Vector(Dimension\_of\_Vector):

vector = [] # Initialize an empty list to store vector elements

for i in range(Dimension\_of\_Vector):

vector.append(float(input(f"Enter the {i}th element of the vector: ")))

return vector

function Manhattan(vector\_A, vector\_B):

manhattan\_distance = 0

for i in range(len(vector\_A)):

manhattan\_distance += abs(vector\_B[i] - vector\_A[i])

return manhattan\_distance

question-2 first we create a class called manhattan which takes vector a and vector b as input fro m the user then we initialize distance =0 . we iterate through length of vector\_A to find the length and get every element and its dimensions. we use the formula and get the Manhattan distance then store it in a variable and return it.

Pseudo code=

function calculate\_Manhattan\_distance(vectorA, vectorB):

distance = 0

for i from 0 to length(vectorB) - 1:

absolute\_difference = absolute\_value(vectorB[i] - vectorA[i])

distance = distance + absolute\_difference

return distance

to find nearest neighbour- create a function for nearest neighbour .one easy way is to calculate Manhattan distance and sort it after that we get corresponding target labels of k nearest neighbours.

Pseudo code-

Question 3:label encoding

We take a function called label encoding with input data from user.seperate it into label encoding

We use dictionaries to map categories to numeric labels. Then we convert data to numeric values

Then we use function to covert the data and print numerical data

Pseudo code=

function label\_encoding(data):

unique\_categories = set(data)

encoding\_dict = {}

for i from 0 to length(unique\_categories) - 1:

encoding\_dict[unique\_categories[i]] = i

numerical\_data = []

for category in data:

numerical\_data.append(encoding\_dict[category])

return numerical\_data

categorical\_data = ["dog", "cat", "dog", "bird", "cat", "bird", "dog"]

numerical\_data = label\_encoding(categorical\_data)

print("The variables to be converted: ", categorical\_data)

print("After converting the variables to label encoding data: ")

print(numerical\_data)

question 4-one hot encoding

we create a function to find one hot encoding. We get unique categories for set of data and then

iterate through categories to find unique character. Then we store one hot encoding for the

category in the dictionary

convert categorical data to one hot encoded numeric data

print the result

pseudo code-

function one\_hot\_encoding(data):

unique\_categories = set(data)

encoding\_dict = {}

for category in unique\_categories:

one\_hot\_encoding = []

for unique\_category in unique\_categories:

if category == unique\_category:

one\_hot\_encoding.append(1)

else:

one\_hot\_encoding.append(0)

encoding\_dict[category] = one\_hot\_encoding in the dictionary

one\_hot\_encoded\_data = []

for category in data:

one\_hot\_encoded\_data.append(encoding\_dict[category])

return one\_hot\_encoded\_data

categorical\_data = ["dog", "cat", "dog", "bird", "cat", "bird", "dog"]

data

numerical\_data = one\_hot\_encoding(categorical\_data)

print("The variables to be converted: ", categorical\_data)

print("After converting the variables to one-hot encoding data: ")

for item