TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA Công Nghệ Thông Tin
BỘ MÔN: Công Nghệ Phần Mềm

ĐỀ THI VÀ BÀI LÀM

Tên học phần: Trí tuệ nhân tạo
Mã học phần: Hình thức thi: *Tự luận có giám sát*
Đề số: **00004** Thời gian làm bài: 70 phút *(không kể thời gian chép/phát đề)*
Được sử dụng tài liệu khi làm bài.

Họ tên: Trương Thị Mỹ Duyên **Lớp**:19TCLC-DT4 **MSSV**:102190160

Sinh viên làm bài trực tiếp trên tệp này, lưu tệp với định dạng MSSV_HọTên.pdf và nộp bài thông qua MSTeam:

Câu 1 (*4 điểm*): Cho tập dữ liệu input.csv với 80 mẫu dữ liệu, mỗi mẫu có 4 đặc trưng (chiều dài đài hoa, chiều rộng đài hoa, chiều dài cánh hoa, chiều rộng cánh hoa) và tên loài hoa tương ứng.

a) *(3 điểm)* Hãy viết chương trình phân loại hoa sử dụng Logistic Regression. Nêu rõ mô hình thức phân loại trong chương trình như thế nào (Ví dụ: có bao nhiêu tế bào nơ-ron, mỗi nơ-ron phụ trách công việc gì, làm sao để phân loại,…)?

 # Trả lời: Dán code vào bên dưới

bai1_duyen2.py > ...

```
61
62  if __name__ == '__main__':
63      print("ok")
64      # data:
65      data = pd.read_csv('input.csv')
66      conditions = ["Iris-setosa", "Iris-versicolor"]
67      values = [0, 1]
68      data["Labeling"] = data["Labeling"].replace(conditions, values)
69      X = data.drop(["Labeling"], axis=1)
70      y = data["Labeling"]
71      data_test = pd.read_csv('output_04 .csv')
72      X_test = data_test.drop(["Labeling"], axis=1)
73      y_test = data_test["Labeling"]
74      # extended data
75      Xbar = np.concatenate((np.ones((1, X.shape[1])), X), axis=0)
76      epsilon = .05
77      d = Xbar.shape[0]
78      w_init = np.random.randn(d, 1)
79      w = my_logistic_sigmoid_regression(Xbar, y, w_init, epsilon)
80      print(w[-1].T)
81      yi = sigmoid(np.dot(w[-1].T, Xbar))
82      print(yi)
83      print(yi.shape)
84
85      yi = sigmoid(np.dot(w[-1].T, X_test))
86      for i in yi:
87          if i > 0.5:
88              print("C: 1")
89          else:
90              print("C: 0")
```

Trả lời: Mô tả mô hình phân loại bằng hình ảnh hoặc bằng lời.

Input: Cho dữ liệu data test vào
Output: Nhận diện đó là label Iris-setosa hay Iris-versicolor

b) *(1 điểm)* Hãy thực thi chương trình và cho biết nhãn của 10 mẫu dữ liệu trong output4.csv

Trả lời: Dán code thực thi thành công

Trả lời: Dán kết quả nhãn ứng với 10 mẫu dữ liệu

Câu 2 (2 *điểm*): Cho không gian Oxyz với 6 điểm có tọa độ tương ứng (0,2,2), (2,2,0) (3,3,0),(3,2,2),(2,3,2) và (2,3,1).

a) *(1 điểm)* Viết hàm thực thi thuật toán *k*-means

 # Trả lời: Dán code vào bên dưới

```
#chon ngau nhien trong X , k trung tam
def random_centers(X,k):
    return X[np.random.choice(X.shape[0],k,replace=False)]

def assign_labels_from_center(X,centers):
    # tinh toan khoang cach giua cac diem so voi trung tam
    D = cdist(X,centers)
    # tra ve phan cum dua tren khoang cach gan nhat cua cac diem so voi cac trung tam
    return np.argmin(D,axis=1)

def update_new_centers(X,labels,K):
    centers = np.zeros((K,X.shape[1]))
    # K = 3
    for k in range(K):
        # tinh trung binh cong die trug tam moi cua cum k
        centers[k,:] = np.mean(X[k],axis=0)
    return centers

def compire(centers,new_centers):
    return set([tuple(a) for a in centers])==set([tuple(a) for a in new_centers])
```

```
def kmeans(X,k):
    # khoi tao center
    centers = [random_centers(X,k)]
    # khoi tao nhan
    labels = []
    while True:
        # cap nhat phan cum
        labels.append(assign_labels_from_center(X,centers[-1]))
        # cap nhat new center sau khi phan cum moi
        new_centers = update_new_centers(X,labels[-1],k)
        if compire(centers[-1],new_centers):
            break
        centers.append((new_centers))
    return (centers,labels)
```

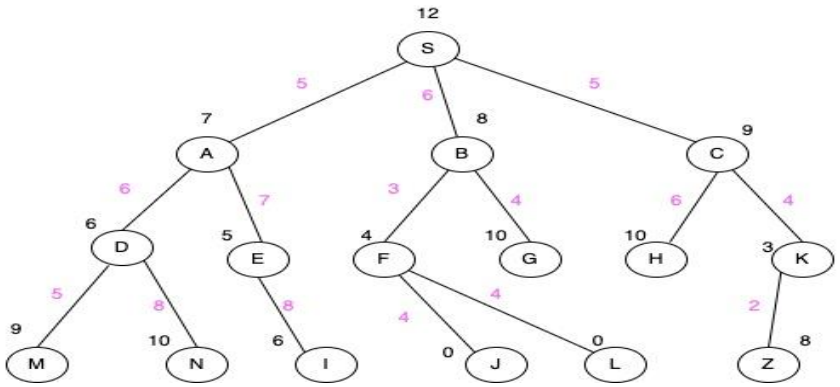
```python

if __name__ == "__main__":

    X = [
        (0,2,2),
        (2,2,0),
        (3,3,0),
        (3,2,2),
        (2,3,2),
        (2,3,1)
    ]
    X = np.asarray(X)
    k = 2
    (centers,labels) = kmeans(X,k)
    print('centers: ',centers)
    print('labels: ',labels)
```

b) *(1 điểm)* Nếu sử dụng thuật toán *k*-means với k = 3 thì kết quả phân nhóm sẽ như thế nào? (các điểm thuộc mỗi nhóm, trọng tâm của mỗi nhóm).

 # Trả lời: viết câu trả lời vào bên dưới

Nhóm 1:
Center: [0,2,2]
Các điểm: [(3,2,2),(2,3,2)]
Nhóm 2:
Center: (3,2,2)
Các điểm: (3,3,0),(2,3,1)
Nhóm 3:
Center: (2,2,0)
Các điểm: (0,2,2),(2,2,0)

Câu 3 (*4 điểm*): Cho cây G = (V,E) như hình vẽ với V là tập đỉnh và E là tập cạnh, trọng số tại đỉnh là hàm ước lượng khoảng cách từ đỉnh đó đến trạng thái đích(giá trị các nhỏ thì càng gần trạng thái đích), trọng số trên cạnh thể hiện chi phí phải trả khi đi qua cạnh.



a) *(2 điểm)* Hãy viết đoạn code biểu diễn đồ thị trên bằng cách khởi tạo tập đỉnh V, tập cạnh E, trọng số trên đỉnh và trọng số trên cạnh.

 # Trả lời: Dán code vào bên dưới

GiaiDeCuoiKi > bai1a.py > Node > __init__ > cost

```python
import heapq

class Node:
    def __init__(self, name,cost=0):

        self.name = name
        self.cost = cost
        self.children = []
        self.parent = []

    def get_name(self):
        return self.name
    def get_data(self):
        return self.name, self.cost

class Tree:
    def __init__(self):
        self.nodes = []
        self.edges = []

    def add_node(self,node):
        self.nodes.append(node)

    def add_nodes(self,nodes):
        for node in nodes:
            self.nodes.append(node)


    def get_index(self, node):
        for i, n in enumerate(self.nodes):
            if n.get_name() == node.get_name():
                return i
        return -1
```

```
28
29      def get_index(self, node):
30          for i, n in enumerate(self.nodes):
31              if n.get_name() == node.get_name():
32                  return i
33          return -1
34
35      def add_edges(self, tuple_edges):
36          for t in tuple_edges:
37              start_label = t[0]
38              end_label = t[1]
39              index_start_label = self.get_index(Node(start_label))
40              index_end_label = self.get_index(Node(end_label))
41              self.nodes[index_start_label].children.append(self.nodes[index_end_label])
42              self.nodes[index_end_label].parent.append(self.nodes[index_start_label])
43              self.edges.append((self.nodes[index_start_label], self.nodes[index_end_label]))
44
45      def show_nodes(self):
46          return [print(node.get_data(),end=",") for node in self.nodes]
47      def get_edge(self, start_node, end_node):
48          for edges in self.edges:
49              print("(",edges[0].get_name(),",", edges[1].get_name(),")",end="\t")
50
```

```python
if __name__ == "__main__":
    tree = Tree()
    NodeS = Node("S",3)
    NodeA = Node("A",4)
    NodeB = Node("B",5)
    NodeC = Node("C",6)
    NodeD = Node("D",7)
    NodeE = Node("E",8)
    NodeF = Node("F",9)
    NodeG = Node("G",1)
    NodeH = Node("H",1)

    tree = Tree()
    tree.add_nodes([
        Node("S",12),
        Node("A", 7),
        Node("B", 8),
        Node("C", 9),
        Node("D", 6),
        Node("E", 5),
        Node("F", 4),
        Node("G", 10),
        Node("H", 10),
        Node("K", 3),
        Node("M", 9),
        Node("N", 10),
        Node("I", 6),
        Node("J", 0),
        Node("L", 0),
        Node("Z", 8),
```

```
GiaiDeCuoiKi > bai1a.py > Node > __init__
64      tree.add_nodes([
65          Node("S",12),
66          Node("A", 7),
67          Node("B", 8),
68          Node("C", 9),
69          Node("D", 6),
70          Node("E", 5),
71          Node("F", 4),
72          Node("G", 10),
73          Node("H", 10),
74          Node("K", 3),
75          Node("M", 9),
76          Node("N", 10),
77          Node("I", 6),
78          Node("J", 0),
79          Node("L", 0),
80          Node("Z", 8),
81      ])
82      tree.add_edges([
83          ("S", "A"),
84          ("S", "B"),
85          ("S", "C"),
86
87          ("A", "D"),
88          ("A", "E"),
89
90          ("B", "F"),
91          ("B", "G"),
92
93          ("C", "H"),
94          ("C", "K"),
95
96          ("D", "M"),
```

```
PS D:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu> python -u "d:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu\GiaiDeCuoiKi\bai1a.py"
Tap dinh:
('S', 12),('A', 7),('B', 8),('C', 9),('D', 6),('E', 5),('F', 4),('G', 10),('H', 10),('K', 3),('M', 9),('N', 10),('I', 6),('J', 0),('L', 0),('Z', 8),
Tap canh:
( S , A )      ( S , B )      ( S , C )      ( A , D )      ( A , E )      ( B , F )      ( B , G )      ( C , H )      ( C , K )      ( D , M )
 D , N )       ( E , I )      ( F , J )      ( F , L )      ( K , Z )
PS D:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu>
```

b) *(2 điểm)* Hãy viết chương trình sử dụng thuật toán **A*** để tìm đường đi từ đỉnh "S" đến các đỉnh có trọng số trên đỉnh bằng 0. Trong chương trình, hãy in ra thứ tự đỉnh khám phá trong quá trình tìm kiếm.

Trả lời: Dán code vào bên dưới

```
explored: S A C B K F G Z H D E J
PS D:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu> python -u "d:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu\GiaiDeCuoiKi\bai1_AStar.py"
From "S" to "J":
{'label': 'S', 'cost': 0, 'goal cost': 12}
{'label': 'A', 'cost': 5, 'goal cost': 7}
{'label': 'C', 'cost': 5, 'goal cost': 9}
{'label': 'B', 'cost': 6, 'goal cost': 8}
{'label': 'K', 'cost': 9, 'goal cost': 3}
{'label': 'F', 'cost': 9, 'goal cost': 4}
{'label': 'G', 'cost': 10, 'goal cost': 10}
{'label': 'Z', 'cost': 11, 'goal cost': 8}
{'label': 'H', 'cost': 11, 'goal cost': 10}
{'label': 'D', 'cost': 11, 'goal cost': 6}
{'label': 'E', 'cost': 12, 'goal cost': 5}
{'label': 'J', 'cost': 13, 'goal cost': 0}
explored: S
explored: S A
explored: S A C
explored: S A C B
explored: S A C B K
explored: S A C B K F
explored: S A C B K F G
explored: S A C B K F G Z
explored: S A C B K F G Z H
explored: S A C B K F G Z H D
explored: S A C B K F G Z H D E
explored: S A C B K F G Z H D E J
From "S" to "L":
```

```
From "S" to "L":
{'label': 'S', 'cost': 0, 'goal cost': 12}
{'label': 'A', 'cost': 5, 'goal cost': 7}
{'label': 'C', 'cost': 5, 'goal cost': 9}
{'label': 'B', 'cost': 6, 'goal cost': 8}
{'label': 'K', 'cost': 9, 'goal cost': 3}
{'label': 'F', 'cost': 9, 'goal cost': 4}
{'label': 'G', 'cost': 10, 'goal cost': 10}
{'label': 'Z', 'cost': 11, 'goal cost': 8}
{'label': 'H', 'cost': 11, 'goal cost': 10}
{'label': 'D', 'cost': 11, 'goal cost': 6}
{'label': 'E', 'cost': 12, 'goal cost': 5}
{'label': 'J', 'cost': 13, 'goal cost': 0}
{'label': 'L', 'cost': 13, 'goal cost': 0}
explored: S
explored: S A
explored: S A C
explored: S A C B
explored: S A C B K
explored: S A C B K F
explored: S A C B K F G
explored: S A C B K F G Z
explored: S A C B K F G Z H
explored: S A C B K F G Z H D
explored: S A C B K F G Z H D E
explored: S A C B K F G Z H D E J
explored: S A C B K F G Z H D E J L
PS D:\Dai hoc\Nam3\HocKi2\TriTueNhanTao\baitapcode_Hieu>
```

Trả lời: Dán kết quả thực thi vào bên dưới

Đà Nẵng, ngày 5 tháng 06 năm 2022

GIẢNG VIÊN BIÊN SOẠN ĐỀ THI

TRƯỞNG BỘ MÔN
(đã duyệt)