DATA SCIENCE AND MACHINE LEARNING (MSC)

DAMA60: Algorithmic Techniques and Systems for Data Science and Machine Learning

Academic Year: 2023–2024

#3 Written Assignment		
Submission Deadline	**Wed, 14 February 2024, 11:59 PM**	
Name/Student Id	**Kritikos Antonios**	**std157978**

Remarks

The deadline is definitive.

An indicative solution will be posted online along with the return of the graded assignments.

The assignment is due via the STUDY submission system. **You are expected to turn in a document (.DOC, .ODT, .PDF) and a file containing a Python program:**

- 1 document file (this document) with the answers to all the questions, along with the Python code snippets for Topic 5.
- 1 file with the complete Python program that answers Topic 5.

You should not make any changes in the written assignment file other than providing your own answers. You should also type all of your answers into Word and not attach any handwritten notes as pictures into your work. Make sure to name all the files (DOC file and Python program file) with **your last name first followed by a dash symbol and the names of each component at the end**. For example, for the student with last name Aggelou the files should be named as follows: Aggelou-HW3.doc (or Aggelou-HW3.pdf), Aggelou-Topic5.py. Please ensure that all Python code you submit for this assignment opens, displays and executes successfully within the IDLE environment — the official Integrated Development and Learning Environment provided with Python 3. This consistency in the execution environment is necessary for a thorough and equitable evaluation of your work.

Topic	Points	Grades
1. Online Quiz	40	
2. Ranking in Graphs	15	
3. PCY algorithm	15	
4. Clustering	15	
5. Python	15	
TOTAL	**100**	**/100**

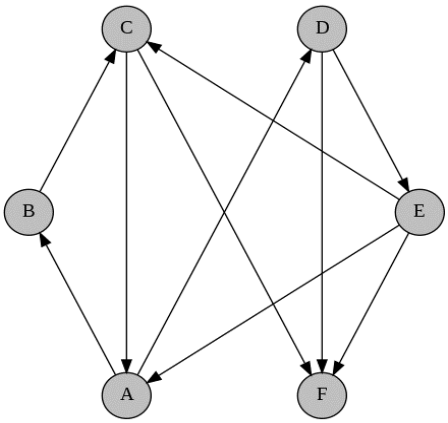
Topic 1: Online Quiz

(40 points) Complete the corresponding online quiz available at:
https://study.eap.gr/mod/quiz/view.php?id=26997

You have one effort and unlimited time to complete the quiz, up to the submission deadline.

Topic 2: Ranking in Graphs

(15 total points) Consider a network of 6 nodes presented below:



(a) (2 points) Fill in the placeholders (?) of the adjacency matrix ***A*** of the above graph, such that $A_{xy} = 1$ if there is an edge between the node x and the node y, and $A_{xy} = 0$ otherwise.

Answer:

	A	B	C	D	E	F
A	0	1	0	1	0	0
B	0	0	1	0	0	0
C	1	0	0	0	0	**1**
D	0	0	0	0	**1**	1
E	1	0	1	**0**	0	1
F	0	0	**0**	0	0	0

(b) (3 points) Fill in the placeholders (?) of the out-degree matrix ***D*** of the given directed graph, such that out-degree of a node is the number of edges pointing away from it.

Answer:

	A	B	C	D	E	F
A	**2**	0	0	0	0	0
B	0	**1**	0	0	0	0
C	0	0	**2**	0	0	0
D	0	0	0	**2**	0	0
E	0	0	0	0	**3**	0
F	0	0	0	0	0	**0**

(c) (**5 points**) After removing the dead-end node, recalculate the out-degree matrix D and fill in the placeholders (?) of the transition probability matrix **M** of going from each node to any other adjacent node, such that the probabilities sum up to one in a column-wise way.

Answer:

Out-degree matrix D

	A	**B**	**C**	**D**	**E**
A	**2**	0	0	0	0
B	0	**1**	0	0	0
C	0	0	**1**	0	0
D	0	0	0	**1**	0
E	0	0	0	0	**2**

Transition probability matrix M

	A	**B**	**C**	**D**	**E**
A	0	**0**	**1**	0	0.5
B	**0.5**	0	0	0	0
C	**0**	1	0	0	0.5
D	**0.5**	0	0	0	0
E	0	0	0	1	0

(d) (2 points) Assuming that all the remaining nodes from the previous question (A – E) share the same initial probabilities, calculate by hand the values of the column vector $\boldsymbol{v}$' by executing the simplified PageRank without teleportation for the **2nd iteration** and then fill in the placeholders (?), **using an accuracy of 3 decimal digits**.

Answer:

Iteration 2: $\boldsymbol{v}' = [0.400, 0.150, 0.200, 0.150, 0.100]^T$

(e) (3 points) Calculate by hand the values of vector **v'** by executing the PageRank with teleportation for the **2nd Iteration** for the simplified graph and then fill in the placeholders (?), **using an accuracy of 3 decimal digits**. Set $\beta = 0.80$.

Answer:

Iteration 2: $\boldsymbol{v}' = [0.344, 0.152, 0.216, 0.152, 0.136]^T$

Topic 3: PCY algorithm

(15 total points) In the following table, a collection of 10 transactions is given. Each transaction contains a set of items derived from seven distinct items, enumerated from 1 to 7. The support threshold is set to 5, thus support values greater or equal to 5 are accepted as frequent.

Transactions	Items
T1	1, 2, 5
T2	2, 3, 6
T3	3, 4, 5
T4	1, 3, 5
T5	2, 4, 7
T6	1, 5, 6
T7	2, 3, 4
T8	2, 4, 5, 7
T9	3, 5, 7
T10	2, 4

(a) (2 Points) Compute the support for each item and each pair of items and fill in the placeholders (**?**).

Answer:

Item	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Support	**3**	**6**	**5**	**5**	**6**	**2**	**3**

Pair	**1, 2**	**1, 3**	**1, 5**	**1, 6**	**2, 3**	**2, 4**	**2, 5**
Support	**1**	**1**	**3**	**1**	**2**	**4**	**2**

Pair	**2, 6**	**2, 7**	**3, 4**	**3, 5**	**3, 6**	**3, 7**	**4, 5**
Support	**1**	**2**	**2**	**3**	**1**	**1**	**2**

Pair	**4, 7**	**5, 6**	**5, 7**
Support	**2**	**1**	**2**

(b) (6 Points) Using a hash table consisting of seven (7) buckets, with each pair $\{i, j\}$ hashed to bucket $(i \times j)\ mod\ 7$, calculate the total support of every bucket.

Answer:

Bucket	0	1	2	3	4	5	6
Support	7	7	2	3	1	6	5

(c) (3 points) Which buckets are frequent?

Answer:

Frequent buckets are: 0, **1**, **5**, **6.**

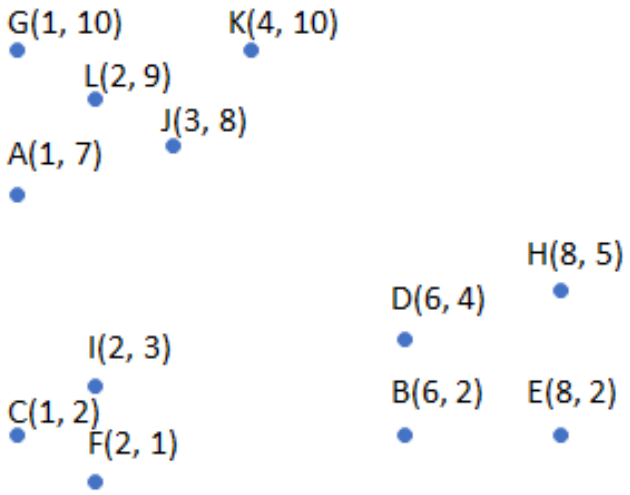
(d) (4 points) Which pairs are finally counted in the 2nd pass of PCY?

Answer:

Candidate pairs are: {2, 3}, {**3**, **4**}, {**3**, **5**}, {**4**, **5**}, {**2**, **4**}

Topic 4: Clustering

(15 total points) Consider the following 12 points in a two-dimensional Euclidean space:



Assuming that the number of clusters is 3 ($k = 3$), we need to initialize a clustering algorithm, by picking points that have a good chance of lying in different clusters. To this end, we are going to pick points that are as far away from one another as possible. After choosing the first point randomly, we sequentially add the point whose minimum distance from the selected points is as large as possible.

(a) (2 Points) If we select A as the first cluster center, which other points will be selected as the other two cluster centers, using the Euclidean Distance as the Distance metric?

Answer:

Initial centroids: A, E, F

(b) (4 Points) After the initial centroids are selected, k-means derives the three clusters as {A, G, J, K, L}, {B, D, E, H} and {C, F, I}. Compute the representation of the cluster as in the BFR Algorithm. That is, compute, SUM and SUMSQ and fill in only the placeholders (**?**).

Answer:

Cluster	Points	N	SUM	SUMSQ
0	A, G, J, K, L	**5**	(**11**,**44**)	(**31**,**394**)
1	B, D, E, H	**4**		(**200**,**49**)
2	C, F, I	**3**		(**9**,**14**)

(c) (6 Points) Compute the variance and standard deviation of each cluster in each of the two dimensions, **using an accuracy of 3 decimal digits**. Fill in only the placeholders (**?**).

Answer:

	Cluster 0		Cluster 1		Cluster 2	
	x_0	y_0	x_1	y_1	x_2	y_2
Variance	1.360					0.667
Standard deviation		1.166	1.000	1.299	0.471	

(d) (3 Points) Assign the following points M(6, 8), N(2, 2), and P(6, 5) as members of one of the previously formed clusters (Cluster 0, Cluster 1 and Cluster 2) or mark it as it belongs to the Retained Set.

To accept a point as a cluster member, the Mahalanobis Distance of the point to the cluster center has to be less than 2 standard deviations away from the specific cluster. If the computed distance is greater than that, the examined point is marked as a retained set. **Use an accuracy of 3 decimal digits.**

Answer:

	Cluster 0	Cluster 1	Cluster 2	Retained Set
A	✓			
B		✓		
C			✓	
…				
M				✓
N			✓	
P		✓		

Topic 5: Python

(15 total points) In this exercise you will use just the fundamental Python library of numpy for simulating the operation of HITS algorithm (hyperlink induced topic search), one of the parts that we focused on during Chapter 5 regarding Link Analysis. To be more specific, we are going to apply our algorithm in a specified network, which is represented by its adjacency matrix via the variable L. Based on that input, we ask you to complete the existing code slots with the commands of your choice, without changing any of the other lines.

(a) (4 points) Complete the 4 missing statements according to the comments given, to extract useful information that characterize the network represented by the numpy variable named L.

Incomplete Python Code:

```
import numpy as np

L = np.array([
              [0, 0, 1, 0, 1],
              [1, 1, 1, 1, 0],
              [0, 0, 0, 0, 1],
              [0, 1, 1, 0, 0],
              [1, 0, 0, 0, 0]
                              ])

# print type of variable L

print(f'Type of variable L is {type(L)}')
print('-' * 24)

# print the number of outgoing edges per Node

for node, val in enumerate(L.sum(axis=1)):
    print(f'Node {node} has {val} outgoing edges')
print('-' * 24)

# print the number of incoming edges per Node
for node, val in enumerate(L.sum(axis=0)):
    print(f'Node {node} has {val} incoming edges')
print('-' * 24)

# check if matrix L is symmetric
if (L == L.T).all():
    print('Matrix L is symmetric')
else:
    print('Matrix L is not symmetric')
print('-' * 24)
```

(b) (**7 points**) Complete the 7 missing statements according to the comments given, to calculate the HITS scores (hubbiness and authority scores) for the nodes in adjacency matrix L.

Incomplete Python Code:

```
dim = L.shape

# create a vector of dimension 5x1 with its points being equal to 1
h_input = np.ones((dim[0], 1))

# the number of iterations is defined here as n_iter
n_iter = 5

# apply the HITS algorithm for n_iter iterations
h_history = []
for iteration in range(0, n_iter):

    a = L.T.dot(h_input) # compute the vector of authorities before scaling
    a = a / max(a) # apply the scaling process
    h = L.dot(a) # compute the vector of hubbiness before scaling
    h = h / max(h) # apply the scaling process

    print(f'iteration: {iteration+1} a: ', np.round(a, 3))
    print(f'iteration: {iteration+1} h: ', np.round(h, 3))
    print('-' * 24)

    # keep in a list called h_history the values of vector h of the
    # current iteration rounded to the 3rd decimal digit
    h_history.append(np.round(h, 3))

    # update the h_input for the next iteration with the current value of h
    h_input = h
```

(c) (2 points) Follow the guidelines on the first page (section Remarks) of this assignment and incorporate the .py file that answers this topic into your submission[1].

Answer:

The code in your .py file with name **Kritikos-Topic5.py** conforms to the module's directions and opens and displays correctly in IDLE.

(d) (2 points) Follow the guidelines on the first page (section Remarks) of this assignment and incorporate the .py file that answers this topic into your submission[1].

Answer:

Your code executes correctly when run from your .py file with name **Kritikos-Topic5.py** using IDLE.

[1] Replace the red question mark in the box below with the name of the file containing your code for this topic.