EXPERIMENT 07

CLASS: TE CMPN A ROLL NO. : 19
NAME: REBECCA DIAS PID: 182027

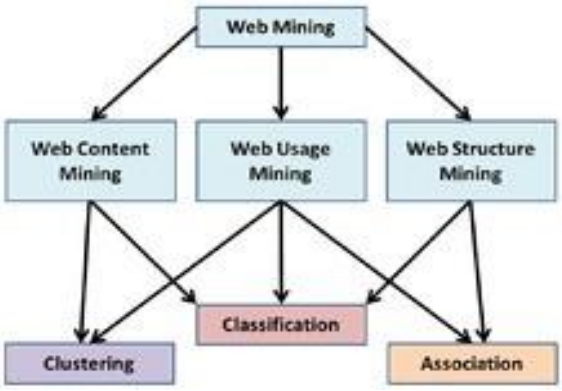
Aim: - Implementation of Page Rank Algorithm.

S/W Requirement: -Python

Theory: -

1. What is Web Mining?

Web mining is the application of data mining techniques to discover patterns from the World Wide Web. It uses automated methods to extract both structured and unstructured data from web pages, server logs and link structures. There are three main sub-categories of web mining. Web content mining extracts information from within a page. Web structure mining discovers the structure of the hyperlinks between documents, categorizing sets of web pages and measuring the similarity and relationship between different sites. Web usage mining finds patterns of usage of web pages.



2. What is Page Rank Algorithm?

Simplified algorithm

Assume a small universe of four web pages: A, B, C and D. Links from a page to itself, or multiple outbound links from one single page to another single page, are ignored. PageRank is initialized to the same value for all pages. In the original form of PageRank, the sum of PageRank over all pages was the total number of pages on the web at that time, so each page in this example would have an initial value of 1. However, later versions of PageRank, and the remainder of this section, assume a probability distribution between 0 and 1. Hence the initial value for each page in this example is 0.25.

The PageRank transferred from a given page to the targets of its outbound links upon the next iteration is divided equally among all outbound links.

If the only links in the system were from pages B, C, and D to A, each link would transfer 0.25 PageRank to A upon the next iteration, for a total of 0.75.

PR(A) = PR(B) + PR(C) + PR(D).

Suppose instead that page B had a link to pages C and A, page C had a link to page A, and page D had links to all three pages. Thus, upon the first iteration, page B would transfer half of its existing value, or 0.125, to page A and the other half, or 0.125, to page C. Page C would transfer all of its existing value, 0.25, to the only page it links to, A. Since D had three outbound links, it would transfer one third of its existing value, or approximately 0.083, to A. At the completion of this iteration, page A will have a PageRank of approximately 0.458.

PR(A) = PR(B)/2 + PR(C)/1 + PR(D)/3

In other words, the PageRank conferred by an outbound link is equal to the document's own PageRank score divided by the number of outbound links L( ).

PR(A) = PR(B)/L(B) + PR(C)/L(C) + PR(D)/L(D)

In the general case, the PageRank value for any page u can be expressed as:

PR(u) = Σu ∈ Bu PR(u)/L(u)

i.e. the PageRank value for a page u is dependent on the PageRank values for each page v contained in the set Bu (the set containing all pages linking to page u), divided by the number L(v) of links from page v. The algorithm involves a damping factor for the calculation of the page rank. It is like the income tax which the govt extracts from one despite paying him itself.

3. Explain its relevance with SEO:

Page rank Algorithm is used by search engines like google in order to rank webpages in their search engine results. Page rank is a way of measuring the importance of website pages. It also measures relevance, reliability and reputation of site. These aspects are then consolidated, taking into account number and quality of links pointing back to the site.

Implementation: -

```
import numpy as np

class CPageRank(object):
    def __init__(self):
        self.PR = []

    def GetPR(self, IOS, alpha, max_itrs, min_delta):
        N = np.shape(IOS)[0]
        e = np.ones(shape=(N, 1))
        L = [np.count_nonzero(e) for e in IOS.T]
        helps_efunc = lambda ios,l:ios/l
        helps_func  = np.frompyfunc(helps_efunc, 2, 1)
        helpS = helps_func(IOS, L)
        A = alpha*helpS + ((1-alpha)/N)*np.dot(e, e.T)
        for i in range(max_itrs):
            if 0 == np.shape(self.PR)[0]:
                self.PR = np.full(shape=(N,1), fill_value=1.0/N)
            print('Initial PR value table:', self.PR)
            old_PR = self.PR
            self.PR = np.dot(A, self.PR)
            D = np.array([old-new for old,new in zip(old_PR, self.PR)])
            ret = [e < min_delta for  e in D]
            if ret.count(True) == N:
                print('Number of iterations: %d, succeed PR:\n'%(i+1), self.PR)
                break
        return self.PR

def CPageRank_manual():
    R = int(input("Enter the number of rows:"))
    C = int(input("Enter the number of columns:"))

    print("Enter the entries in a single line (separated by space): ")
    entries = list(map(int, input().split()))
    matrix = np.array(entries).reshape(R, C)
    '''
    IOS = np.array([[0, 0, 0, 0, 1],
                    [1, 0, 0, 0, 0],
                    [1, 0, 0, 0, 0],
                    [1, 1, 0, 0, 0],
                    [0, 1, 1, 1, 0]], dtype=float)
    '''
    IOS = np.array(matrix, dtype = float)
    print('Directed Graph Matrix: ')
    print(matrix)
    pg = CPageRank()
    ret = pg.GetPR(IOS, alpha=0.85, max_itrs=100, min_delta=0.0001)
    print('Final PR value:\n', ret)
if __name__=='__main__':
    CPageRank_manual()
```

OUTPUT: -

```
Enter the number of rows:5
Enter the number of columns:5
Enter the entries in a single line (separated by space):
0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0  1 1 1 0
Directed Graph Matrix:
[[0 0 0 0 1]
 [1 0 0 0 0]
 [1 0 0 0 0]
 [1 1 0 0 0]
 [0 1 1 1 0]]
Initial PR value table: [[0.2]
 [0.2]
 [0.2]
 [0.2]
 [0.2]]
Initial PR value table: [[0.2]
 [0.0866666666666667]
 [0.0866666666666667]
 [0.1716666666666667]
 [0.4550000000000007]]
Initial PR value table: [[0.4167500000000001]
 [0.0866666666666667]
 [0.0866666666666667]
 [0.12350000000000004]
 [0.2864166666666667]]
Initial PR value table: [[0.2734541666666667]
 [0.1480791666666667]
 [0.1480791666666667]
 [0.18491250000000006]
 [0.2454750000000001]]
  Initial PR value table: [[0.2386537500000001]
   [0.10747868055555558]
   [0.10747868055555558]
   [0.17041232638888895]
   [0.37597656250000017]]
  Initial PR value table: [[0.34958007812500014]
   [0.09761856250000006]
   [0.09761856250000006]
   [0.14329700173611118]
   [0.311885795138889]]
  Initial PR value table: [[0.29510292586805564]
   [0.1290476888020834]
   [0.1290476888020834]
   [0.17053557786458343]
   [0.2762661186631946]]
  Initial PR value table: [[0.26482620086371544]
   [0.11361249566261579]
   [0.11361249566261579]
   [0.16845776340350124]
   [0.33949104440755223]]
  Initial PR value table: [[0.3185673877464194]
   [0.10503409024471941]
   [0.10503409024471941]
   [0.15331940090133112]
   [0.3180450308628113]]
  Initial PR value table: [[0.30033827623338966]
   [0.12026075986148552]
   [0.12026075986148552]
   [0.1649002482154913]
   [0.2942399558281487]]
  Initial PR value table: [[0.28010396245392644]
   [0.11509584493279378]
   [0.11509584493279378]
   [0.1662066678739251]
   [0.32349767980656163]]
```

```
Initial PR value table: [[0.30497302783557745]
 [0.10936278936194586]
 [0.10936278936194586]
 [0.1582785234583832]
 [0.3180228699821484]]
Initial PR value table: [[0.30031943948482615]
 [0.11640902455341366]
 [0.11640902455341366]
 [0.16288821003224063]
 [0.3039743013761067]]
Initial PR value table: [[0.2883781561696907]
 [0.1150905078540341]
 [0.1150905078540341]
 [0.16456434328923492]
 [0.31687648483300696]]
Initial PR value table: [[0.299345012108056]
 [0.11170714424807907]
 [0.11170714424807907]
 [0.16062061008604359]
 [0.3166200893097431 7]]
Initial PR value table: [[0.2991270759132817]
 [0.11481442009728256]
 [0.11481442009728256]
 [0.16228995640271618]
 [0.3089541274894379]]
Initial PR value table: [[0.2926110083660223]
 [0.11475267150876318]
 [0.11475267150876318]
 [0.16354880005010827]
 [0.314334848566344]]
Initial PR value table: [[0.29718462128139245]
 [0.11290645237037303]
 [0.11290645237037303]
 [0.16167633776159737]
 [0.3153261362162651 5]]
Initial PR value table: [[0.2980272157838254]
 [0.11420230936306124]
 [0.11420230936306124]
 [0.1621875516204698]
 [0.3113806138695834]]
Initial PR value table: [[0.2946735217891459]
 [0.11444104447208392]
 [0.11444104447208392]
 [0.16297702595138494]
 [0.31346736331530245]]
Initial PR value table: [[0.2964472588180071]
 [0.11349083117359136]
 [0.11349083117359136]
 [0.16212827507422703]
 [0.3144428037605842 6]]
Initial PR value table: [[0.29727638319649663]
 [0.11399338999843539]
 [0.11399338999843539]
 [0.1622269932472117]
 [0.312509843559422]]
Initial PR value table: [[0.29563336702550874]
 [0.11422830857234077]
 [0.11422830857234077]
 [0.1626754993216758]
 [0.3132345165081351]]
Initial PR value table: [[0.29624933903191486]
 [0.1137627873238942]
 [0.1137627873238942]
 [0.16230981846713902]
 [0.313915267853159]]
Initial PR value table: [[0.2968279776751852]
 [0.11393731272570926]
 [0.11393731272570926]
 [0.16228649733836428]
 [0.3130108995350333 6]]
```

```
Initial PR value table: [[0.2960592646047844]
 [0.11410126034130252]
 [0.11410126034130252]
 [0.16252461824972897]
 [0.313213596462889]]
Initial PR value table: [[0.2962315569934557]
 [0.11388345830468728]
 [0.11388345830468728]
 [0.16237649394974088]
 [0.3136250324474304]]
Initial PR value table: [[0.29658127758031594]
 [0.1139322744814792]
 [0.1139322744814792]
 [0.16233274426097127]
 [0.3132214291957561]]
Initial PR value table: [[0.2962382148163927 3]
 [0.11403136198108957]
 [0.11403136198108957]
 [0.16245257863571824]
 [0.3132464825857116]]
Number of iterations: 29, succeed PR:
 [[0.2962595101978549]
 [0.11393416086464467]
 [0.11393416086464467]
 [0.16239748970660772]
 [0.31347467836624976]]
Final PR value:
 [[0.2962595101978549]
 [0.11393416086464467]
 [0.11393416086464467]
 [0.16239748970660772]
 [0.31347467836624976]]
```

Conclusion: -

Rebecca Dias
182027/19
TE CMPNA

DWM

- Summary of Experiment
  In this experiment we learnt about Page Rank algorithm. We learnt what web mining is and what is the relevance of page Rank and SEO. We implemented a page rank algorithm in python which finds the rank of each web page for a directed graph.

- Importance of Experiment
  Page rank is important because its one of the factors a search engine like Google takes into account when it decides which results to show at the top of its search engine listings - where they can be easily seen. As the algorithm is based on links, the content which is more important factor for the user is neglected.

- Application of Experiment: Page Rank has various applications of which two of them are listed below
  i) Ranking Tweets in twitter
  ii) Recommendation systems, finding suggestions for each user based on their watch history and their rating.