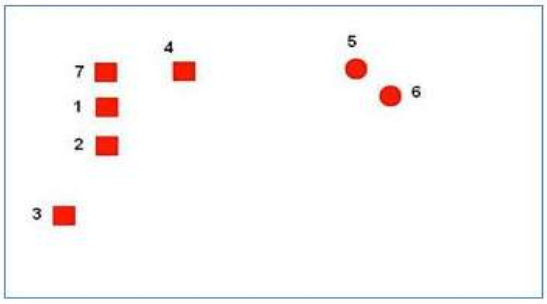
# CS-422 Assignment 5

## Name : Dhayalini Nagaraj

## AID : A20359686

**I Clustering**

**1) Given this example of a small set of points, find three initial centroids so that kMeans with k=3 converges to a clustering with an empty cluster. Note that the initial centroids do not have to be members of the set of points. Show the kMeans steps (Do something simple. For example, mark the location of centroids and circle the points for each centroid for each iteration). Explain your example in detail in your own words.**




K-means clustering is a Partitional clustering approach. Each cluster is associated with a centroid (center point) and each point is assigned to the cluster with the closest centroid. Number of clusters, K, must be specified.

The K-means algorithm involves randomly selecting K initial centroids where K is a user defined number of desired clusters. Each point is then assigned to a closest centroid and the collection of points close to a centroid form a cluster. The centroid gets updated according to the points in the cluster and this process continues until the points stop changing their clusters. A high-level representation of the K-means algorithm

Select $K$ points as the initial centroids.
**repeat**
  Form $K$ clusters by assigning all points to the closest centroid.
  Recompute the centroid of each cluster.
**until** The centroids don't change

Initial centroids are often chosen randomly. Clusters produced vary from one run to another. The centroid is the mean of the points in the cluster. 'Closeness' is measured by Euclidean distance, cosine similarity, correlation, etc.,.

**Choosing initial centroids**

**Randomly**: Different runs of the algorithm can produce poor results when the initial centriods are choosing randomly. It is important to realize that the choice of the initial centroid has a huge effect on the final result.

**Solutions to this problem**

- Multiple Runs: To perform multiple runs, each with a different set of randomly chosen initial centriods and then selects the set of clusters with the minimum SSE. The problem with this method is that it doesn't always work.

- In general, if there are K clusters and each cluster has n points, then the probability, P, of selecting one initial centroid from each cluster is given by the following equation. P = number of ways to select one centroid from each cluster/number of ways to select K centroids

- Take a sample of points and cluster them using a hierarchical clustering technique and then K clusters are extracted from this clustering and the centriods are then used as the initial centriods for the K-means algorithm. The downside to this method is that it is only works if the sample is relatively small and K is relatively small in comparison to the sample size.

- Select the first point at random for the first centroid and then for all other initial centriods (k-1) select a point that is farthest from any of the initial centroids that have already been selected. By doing this we obtain a set of initial centroids that is randomly selected and separated. The down side to this method is that we could end up selecting an outlier as a centroid.

- Upgrading the algorithm and using the bisecting K-means algorithm which is less susceptible to initialization issues.

**Handling Empty Clusters**

Generally K-Means algorithm doesn't always guarantee a global minimum solution . We can obtain global minimum solution only by making good choice of intial set of centroids in the cluster.

One of the problems with the basic K-means algorithm is that empty clusters can be obtained if no points are allocated to a cluster during the assignment step. If this happens, then a strategy is needed to choose a replacement centroid, since otherwise, the squared error will be larger than necessary. Essentially we will have an output with k-1 cluster. The aim of Simple K-Means algorithm is reduce the total sum of squared errors in a cluster.

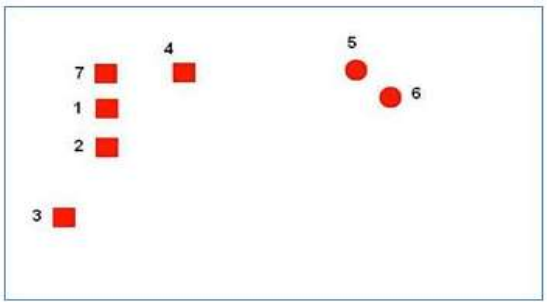
If we need our algorithm to obtain global minimum solution we must do any of the following :

- Choose the point that is farthest away from any current centroid. If nothing else, this eliminates the point that currently contributes most to the total squared error.
- Choose the replacement centroid from the cluster that has the highest SSE. This will typically split the cluster and reduce the overall SSE of the clustering.

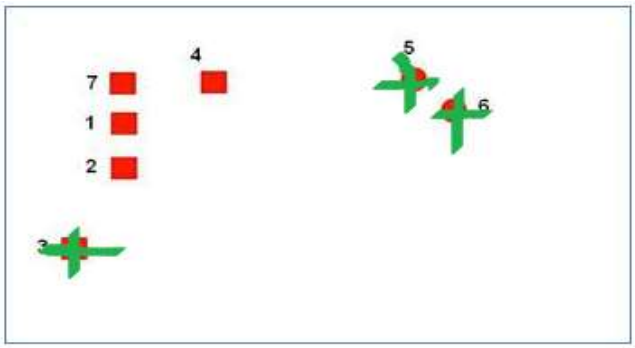
If there are several empty clusters, then this process can be repeated several times.

Let us consider for the given example we want the number of clusters to be 3. In this diagram the shapes of the points are square and circle and it shows that there are actually two natural clusters. The shapes of the points have no meaning for now. Usually we do not know the natural

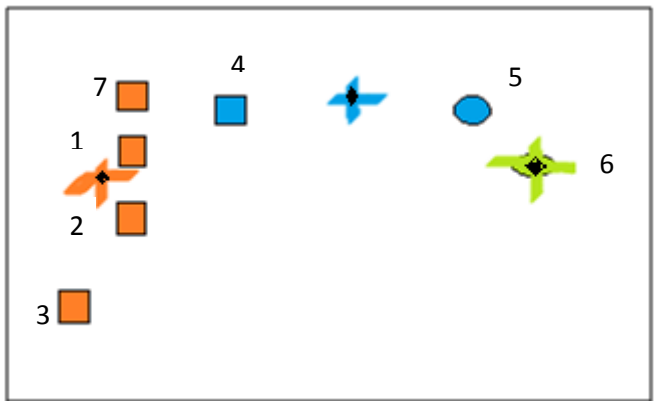
number of clusters in the given data before running K-means clustering algorithm. So to run k-means clustering let us assume that the k=3 for this example.




Let 3, 5 and 6 be our initial centroids. Based on initial centroid choosing methods, I have selected one centroid which is far from other data points 3. Instead of taking a centroid randomly in the R^n space, the data points itself is taken as centroid inorder to minimize the number of iterations and to obtain an optimal solution. Based on the first centroid the other k-1 centroids are selected which is farer from the first centroid. Moreover from visualization of the given example it is also seen one set of points are grouped and the circle points on the other side. Though the shapes doesn't provide any meaning from the visualition it shows that there are two natural clusters. So the three initial centroids are selected in such a way that one of the centroid lies to one side and the other lies to the other side and third one in center or to any one side of the cluster.



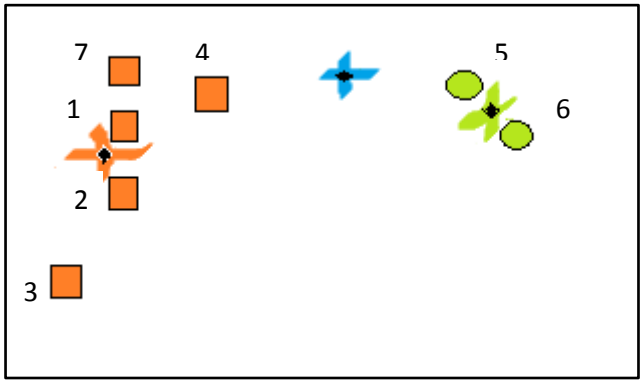

At the first iteration, the points are reassigned to the nearest centroid and then by re-computing, the points to the nearest centroid can be grouped.




After the first iteration, the points 3, 1, 2, and 7 will be in one cluster. 4, 5 will be in next cluster. 6 will be in the last cluster. We are asked to build k means with k=3. The distance between 3 and

4 is larger than the distance between 4 and 5 and the point 4 is assigned to the cluster represented by blue color. The updated cluster centers are shown in the figure.

Analysing the picture we can conclude that the first cluster center is moved nearer to point 4 and the data point 5 is close to the centroid of the last cluster. So in the second iterations the cluster centroids are updated.




Now, the cluster center for the red cluster moved closer to point 4 due to 1, 2, and 7. The cluster center for the blue cluster moved away from 5 due to point 4.

In the second iteration, the cluster center moves towards point 4 and the point 5 is very close to the point 6.As a result of iteration, based on SSE values, the centroid of the red color cluster is nearer to point 4 than the blue cluster. So the point 4 is included in first cluster red and point 5 is included in the third cluster green color because the third cluster centroid is nearer to it than the blue color. The second cluster blue color is empty.

Hence both the data points 4 and 5 which was present earlier in the second blue cluster tends to join red and green clusters. This causes the blue cluster to be empty.

The empty cluster also denotes that it will essentially have output with k-1 clusters. Based on the squared error values the nearer points are grouped into one cluster which produces this empty cluster. Inorder to avoid this empty cluster a replacement centroid with highest SSE value must be selected.

**2) We use SSE(WSS) as the measure of cluster quality and kMeans minimizes it. If there is an empty cluster, can that clustering be the global minimum solution based on RSS? Show all details of you arguments. Use the equation for WSS and how we used it in our discussion in class to prove kMeans convergence. Use your own words.**

No, the K-Means algorithm converges only to a local minimum and not a global minimum. There is no guarantee that a global minimum is reached.

K-means is the most important flat clustering algorithm. The main objective of the k-means algorithm is to minimize the average squared Euclidean distance of documents from their cluster centers where a cluster center is defined as the centroid mean or centroid μ of the documents in a cluster ω: $\mu(\omega) = 1/|\omega| \sum_{x \in \omega} (x)$

**Definition**: Sum of Squared Error (SSE) is a measure used as the objective function to find a good clustering. The error is the distance to the nearest centroid or representative point of a cluster for each point. These terms are then squared and summed to get the sum of squared error measure for k clusters. It is also referred as Residual Sum of Squares (RSS) with the representative points as centroids.

$$\mathrm{RSS}_k = \sum_{\vec{x} \in \omega_k} |\vec{x} - \vec{\mu}(\omega_k)|^2$$

$$\mathrm{RSS} = \sum_{k=1}^{K} \mathrm{RSS}_k$$

$$\mathrm{WSS} = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

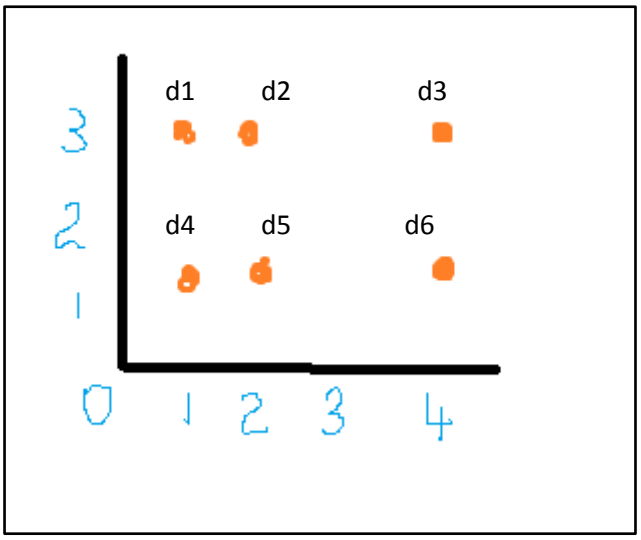
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

K-means algorithm randomly chooses initial centroids. The algorithm then moves the cluster centers around in space in order to minimize RSS. Consider a dataset that have more noise. So we will choose the one of the data points in noise as a centroid. At each iteration reassigning documents to the cluster with the closest centroid; and recomputing each centroid based on the current members of its cluster is done. This iteration is done until a stopping criterion is met. By doing these iterations the RSS is monotonically decreases in each iteration

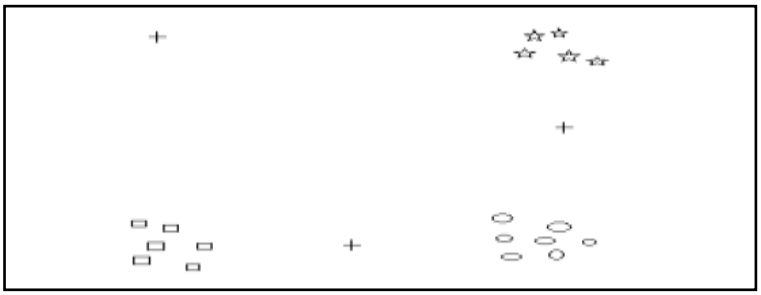
In the reassignment step since each vector is assigned to the closest centroid, the distance it contributes to RSS decreases. It decreases in the recomputation step because the new centroid is the vector for which $\mathrm{RSS}_K$ reaches its minimum. $\mathrm{RSS}_k$ is then decreased when old centroid is replaced by new centroid.

Since there is only a finite set of possible clusterings, a monotonically decreasing algorithm will eventually arrive at a local minimum. Though it arrives at a local minimum it is not guaranteed to be a global minimum. Because it totally depends on the initial centroids selected for the data points.

For example, if a document contain many outliers and are far from other documents then they will not fit into any cluster. So if an outlier is chosen as an initial centroid, then no data points is assigned to that cluster during iterations and it would be a singleton cluster at the end of the iteration. It is a suboptimal result and it is because of the bad choice of intial centroids. (ie) If a centroid is not selected in the noise area and it is far from the other data points, then we would end up in a singleton cluster.

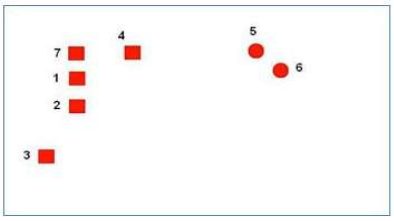



In this figure, for seeds d2 and d5 the k-means converges to {{d1, d2, d3}, {d4, d5, d6}} a suboptimal clustering. For seeds d2 and d3 it converges to {{d1, d2, d4, d5}, {d3, d6}} the global optimum for K=2. Hence from k-means convergence depends on initial seed selection.



From the image we clearly see that there are naturally three clusters. If one of the initial centroid is chosen at an unfavourable location which will result in an empty cluster after iterations. The three clusters will be grouped into two cluster which is only local minimum and not a global minimum.

But if k-means is done for the given example then the solution with empty cluster when k=3 may be a global solution because from the image it is clearly seen that there are two natural clusters and if k=3 one empty cluster will be formed.




Therefore if we achieve a solution with empty cluster we cannot claim that the result is global minimum solution. The resulting local minimum solution may be a global solution but it is not always a global solution. Achieving a global minimum solution is very expensive and is a NP-hard problem.

**II Recommender System**

| | $a$ | $b$ | $c$ | $d$ | $e$ | $f$ | $g$ | $h$ |
|---|---|---|---|---|---|---|---|---|
| $A$ | 4 | 5 | | 5 | 1 | | 3 | 2 |
| $B$ | | 3 | 4 | 3 | 1 | 2 | 1 | |
| $C$ | 2 | | 1 | 3 | | 4 | 5 | 3 |

**Consider the following utility matrix, representing ratings on a 1-5 star scale of eight items a-h by three users A, B, and C. Treat ratings of 3, 4, and 5 as 1 and 1, 2 and blank as 0. Compute the Jaccard, and cosine distances between each pair of users. Which one is better? Explain your answer based on how the measures are computed.**

There is an extensive class of Web applications that involve predicting user responses to options. Such a facility is called a recommendation system. Recommendation systems use a number of different technologies. We can classify these systems into two broad groups.

**Content-based systems** : Recommend items to customer x by examine properties of the items recommended by x.
**Collaborative filtering systems**: Recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users

**Jaccard Similarity** : The Jaccard similarity of sets S and T is $|S \cap T|/|S \cup T|$, that is, the ratio of the size of the intersection of S and T to the size of their union. We shall denote the Jaccard similarity of S and T by SIM(S, T ). We can say two customers are similar if their sets of purchased items have a high Jaccard similarity. Jaccard similarities need not be very high or low to be significant.

**Jaccard Distance**: The Jaccard distance of sets by $d(x, y) = 1 - SIM(x, y)$. That is, the Jaccard distance is 1 minus the ratio of the sizes of the intersection and union of sets x and y.

**The other formula**:
The total number of each combination of attributes for both A and B are specified as follows:

$M_{01}$ = the number of attributes where A was 0 and B was 1
$M_{10}$ = the number of attributes where A was 1 and B was 0
$M_{00}$ = the number of attributes where A was 0 and B was 0
$M_{11}$ = the number of attributes where A was 1 and B was 1

Jaccard similarity = number of 1-1 matches / number of not-both-zero attributes values
$= (M_{11}) / (M_{01} + M_{10} + M_{11})$

**Cosine Distance**: The cosine distance makes sense in spaces that have dimensions, including Euclidean spaces and discrete versions of Euclidean spaces, such as spaces where points are vectors with integer components or boolean (0 or 1) components. Cosine distance = 1- cosine similarity = 1- sim(x,y). This is not a proper cosine distance since it does not hold the property of triangle inequality.

**Cosine similarity** is a measure of similarity between two non zero vectors of an inner product space that measures the cosine of the angle between them.

$$\text{sim}(\boldsymbol{x}, \boldsymbol{y}) = \cos(\boldsymbol{r_x}, \boldsymbol{r_y}) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$$

Let us treat ratings of 3, 4, and 5 as 1 and 1, 2 and blank as 0.

| | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| A | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| B | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| C | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |

**A,B**

Jaccard similarity for A and B, SIM(A, B)= $(M_{11}) / (M_{01} + M_{10} + M_{11})$

= 2/ 1+2+2 =2/5 = 0.4

Jaccard distance = 1- Jaccard similarity of A and B = 1-0.4 =0.6

Cosine similarity =$\sum_i r_{xi}\, r_{yi} \;/ \sqrt{\sum_i r_{xi}^2} \;\cdot \sqrt{\sum_i r_{yi}^2}$

= 0+1+0+1+0+0+0+0/$\sqrt{4}\,\sqrt{3}$ =1/$\sqrt{3}$ = 0.577

Cosine distance = 1-0.577 =0.423

**A,C**

Jaccard similarity for A and C, SIM(A, C)= $(M_{11}) / (M_{01} + M_{10} + M_{11})$

= 2/ 2+2+2 =2/6 = 0.333

Jaccard distance = 1- Jaccard similarity of A and C = 1-0.333 =0.667

Cosine similarity =$\sum_i r_{xi}\, r_{yi} \;/ \sqrt{\sum_i r_{xi}^2} \;\cdot \sqrt{\sum_i r_{yi}^2}$

= 0+0+0+1+0+0+1+0/$\sqrt{4}\,\sqrt{4}$ =1/2 = 0.5

Cosine distance = 1-0.5 = 0.5

**B,C**

Jaccard similarity for B and C, SIM(B, C)= $(M_{11}) / (M_{01} + M_{10} + M_{11})$

= 1/ 3+2+1 =1/6 = 0.1666

Jaccard distance = 1- Jaccard similarity of B and C = 1-0.1666 =0.8334

Cosine similarity =$\sum_i r_{xi}\, r_{yi} \;/ \sqrt{\sum_i r_{xi}^2} \;\cdot \sqrt{\sum_i r_{yi}^2}$

= 0+0+0+1+0+0+0+0/$\sqrt{3}\,\sqrt{4}$ =1/2$\sqrt{3}$ = 0.288

Cosine distance = 1-0.288 = 0.711

- Jaccard similarity ignores the value of the rating.

  If ratings are 1-to-5-stars, put a movie in a customer's set n times if they rated the movie n-stars. Then, use Jaccard similarity for bags when measuring the similarity of customers. The Jaccard similarity for bags B and C is defined by counting an element n times in the intersection if n is the minimum of the number of times the element appears in B and C. In the union, we count the element the sum of the number of times it appears in B and in C.

- Cosine similarity takes the missing values as negative.
  The cosine distance makes sense in spaces that have dimensions, including Euclidean spaces and discrete versions of Euclidean spaces, such as spaces where points are vectors with integer components or boolean (0 or 1) components.

- Jaccard Distance: If we could ignore values in the matrix and focus only on the sets of items rated then this measure would be a good one to choose. However, when utilities are more detailed ratings, the Jaccard distance loses important information.

A distance measure on this space is a function d(x, y) that takes two points in the space as arguments and produces a real number, and satisfies the following axioms:
1. $d(x, y) \geq 0$ (no negative distances).
2. $d(x, y) = 0$ if and only if $x = y$ (distances are positive, except for the distance from a point to itself).
3. $d(x, y) = d(y, x)$ (distance is symmetric).
4. $d(x, y) \leq d(x, z) + d(z, y)$ (the triangle inequality).

**Jaccard distance measure is better.**

Jaccard distance is appropriate when the matrix consists only of 1's and blanks or 0 for "not rated". Here the ratings of the movie is converted into 0 and 1. Moreover Cosine distance works for more general values in the utility matrix. It is often useful to normalize the utility matrix by subtracting the average value (either by row, by column, or both) before measuring the cosine distance. The cosine distance calculated using the formula 1-SIM(x,y) is not a proper distance since it does not have triangle inequality property. Cosine similarity takes missing values as negative. Jaccard similarity should be not be too high or too low. Too high and low similarity may lead to a wrong answer. So for the above problem Jaccard distance measure would be more accurate and it gives the correct distance measure between the users. Lesser the jaccard distance, the users appear closer with similarities.

Jaccard distance (B,C) > Jaccard distance (A,C) >Jaccard distance (A,B)

**III Page Rank**
**• Use the PageRank approach to find influential Twitter users**

**Calculate the PageRank for a selection of four users based on the following four tweets:**
**• user: Tim, tweet: "@Tom Howdy!"**

**• user: Mike, tweet: "Welcome @Tom and @Anne!"**

**• user: Tom, tweet: "Hi @Mike and @Anne!"**

**• user: Anne, tweet: "Howdy!"**

**There are four short tweets generated by four users. The @mentions between users form a directed graph with four nodes and five edges. E.g., the "Tim" node has a directed edge to the "Tom" node.**

**Q: Compute manually the first 3 iterations of the PageRank iterations over this 4 node graph. You should use 0.1 as the probability of teleporting. Show all steps of your calculation, provide details and explanations for them (explain the matrices, the vectors you are using, and the equations). Write down the rank order of the 4 users after on you compute 3 iterations.**

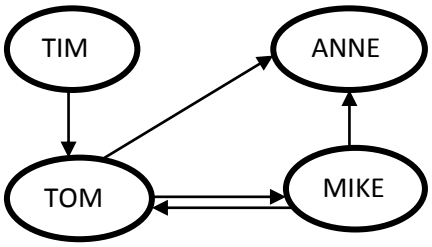
PageRank is an algorithm used by Google Search to rank websites in their search engine results. PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative importance within the set. The algorithm may be applied to any collection of entities with reciprocal quotations and references. The numerical weight that it assigns to any given element E is referred to as the PageRank of E and denoted by PR(E)

There are two main things that need to be noted.

- There should not be dead ends (a node that has no outlinks)
- There should not be any traps (nodes that link to themselves) in the structure.

Let us construct the graph based on the tweets by the various user: The graph has four nodes and five edges.




Let the ranks for users Tim, Tom, Mike and Anne be $r_{Tim}$, $r_{Tom}$, $r_{Mike}$ and $r_{Anne}$ respectively.

The rank vector be represented by matrix r as in the below mentioned order

$$r = \begin{pmatrix} r_{Tim} \\ r_{Tom} \\ r_{Mike} \\ r_{Anne} \end{pmatrix}$$

Here $r_{Tim} + r_{Tom} + r_{Mike} + r_{Anne} = 1$ as $\sum_i r_i = 1$

**Flow Equation**: The flow equation can be derived from the formula where $d_i$ is the number of outgoing edges for node i.

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

The transition matrix M for the given graph is as below.
This matrix M has n rows and columns, if there are n pages. The element $M_{ij}$ in row i and column j has value 1/k if page j has k arcs out, and one of them is to page i. Otherwise $M_{ij}$ is zero.

| | Tim | Tom | Mike | Anne |
|---|---|---|---|---|
| Tim | 0 | 0 | 0 | 0 |
| Tom | 1 | 0 | ½ | 0 |
| Mike | 0 | ½ | 0 | 0 |
| Anne | 0 | ½ | ½ | 0 |

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & ½ & 0 \\ 0 & ½ & 0 & 0 \\ 0 & ½ & ½ & 0 \end{pmatrix}$$

From the graph, we know that, Tim has only one outgoing edge, Tom has two outgoing edges, Mike has two outgoing edges and Anne has no outgoing edges. Anne is a dead end as there no outgoing links from Anne. This kind of scenario is called "dead ends" problem in page rank. In order to avoid this we need to add a constant which is called as probability of teleporting (1 - β).

**PageRank Formula**:

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta)\frac{1}{n}$$

The Google Matrix A can be written as, A=βM + (1-β)/n and therefore r =Ar

The rank vector after a specified number of iterations, v' is given by
v' = M $\beta v_0$ + (1 – β) e/N

Initially, we give equal ranks to all pages.
Hence our initial page rank vector is,

$$v_0 = \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$$

M - transition matrix
N – number of nodes
e - vector of all 1's
1 – β - probability of teleporting
v - page rank estimate

Since there is a dead ends problem, the matrix is not column stochastic. The initial assumptions are still not met.

There are two ways to resolve the dead ends problem:
i) We can either preprocess matrix to remove all dead ends.
ii) Explicitly follow random teleport links with probability 1.0 from dead-ends.

Since by the graph we know that Anne's is a dead end, we can use the first method and make the Anne's column stochastic by replacing the 0's with a constant factor of ¼.
$1 - \beta = 0.1$
Therefore, $\beta = 0.9$

$\beta$ generally lies between 0.8 – 0.9

The new transition matrix M after removing the dead ends is

| | TIM | TOM | MIKE | ANNE |
|---|---|---|---|---|
| TIM | 0 | 0 | 0 | ¼ |
| TOM | 1 | 0 | ½ | ¼ |
| MIKE | 0 | ½ | 0 | ¼ |
| ANNE | 0 | ½ | ½ | ¼ |

**First Iteration:**
$v_0$ is the initial page rank vector with equal importance for all pages.

$v' = M \beta v_0 + (1 - \beta) e/N$

$$= 0.9 \begin{pmatrix} 0 & 0 & 0 & 1/4 \\ 1 & 0 & 1/2 & 1/4 \\ 0 & 1/2 & 0 & 1/4 \\ 0 & 1/2 & 1/2 & 1/4 \end{pmatrix} \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix} + 0.1 \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$$

$$= \begin{pmatrix} 0.05625 \\ 0.39375 \\ 0.16875 \\ 0.28125 \end{pmatrix} + \begin{pmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{pmatrix}$$

$$v' = \begin{pmatrix} 0.08125 \\ 0.41875 \\ 0.19375 \\ 0.30625 \end{pmatrix}$$

**Second Iteration**:

$v'' = M\,\beta v' + (1-\beta)\,e/N$

$$= 0.9 \begin{pmatrix} 0 & 0 & 0 & 1/4 \\ 1 & 0 & 1/2 & 1/4 \\ 0 & 1/2 & 0 & 1/4 \\ 0 & 1/2 & 1/2 & 1/4 \end{pmatrix} \begin{pmatrix} 0.08125 \\ 0.41875 \\ 0.19375 \\ 0.30625 \end{pmatrix} + 0.1 \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$$

$$= \begin{pmatrix} 0.06890 \\ 0.22921 \\ 0.25734 \\ 0.34453 \end{pmatrix} + \begin{pmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{pmatrix}$$

$$v'' = \begin{pmatrix} 0.0939 \\ 0.2542 \\ 0.2823 \\ 0.3695 \end{pmatrix}$$

**Third Iteration**:

$v''' = M\,\beta v'' + (1-\beta)\,e/N$

$$= 0.9 \begin{pmatrix} 0 & 0 & 0 & 1/4 \\ 1 & 0 & 1/2 & 1/4 \\ 0 & 1/2 & 0 & 1/4 \\ 0 & 1/2 & 1/2 & 1/4 \end{pmatrix} \begin{pmatrix} 0.0939 \\ 0.2542 \\ 0.2823 \\ 0.3695 \end{pmatrix} + 0.1 \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$$

$$= \begin{pmatrix} 0.0831 \\ 0.2946 \\ 0.1975 \\ 0.3245 \end{pmatrix} + \begin{pmatrix} 0.025 \\ 0.025 \\ 0.025 \\ 0.025 \end{pmatrix}$$

$$v''' = \begin{pmatrix} 0.1081 \\ 0.3197 \\ 0.2225 \\ 0.3495 \end{pmatrix}$$

Thus after three iterations the rank of Anne is more i.e. Anne is the most influential user, followed by Tom, Mike and Tim.

Anne rank 0.3495 > Tom rank 0.3197 > Mike 0.2225 > Tim rank 0.1081

The rank order for the users is $r_{Anne} > r_{Tom} > r_{Mike} > r_{Tim}$ .

| Rank Order | User | Score |
|---|---|---|
| 1 | Anne | 0.3495 |
| 2 | Tom | 0.3197 |
| 3 | Mike | 0.2225 |
| 4 | Tim | 0.1081 |

Note : Rank vector represented by matrix v', v'', v''' and its sum $r_{Tim} + r_{Tom} + r_{Mike} + r_{Anne}$ tends to 1.

**Reference:**


K-means convergence and Recommendation Systems topics are referred from the below links.

http://nlp.stanford.edu/IR-book/html/htmledition/k-means-1.html

http://www.dsc.ufcg.edu.br/~sampaio/Livros/IntroductionToInformationRetrieval/CapituloACapitulo/16flat.pdf

http://infolab.stanford.edu/~ullman/mmds/ch3.pdf

http://infolab.stanford.edu/~ullman/mmds/ch9.pdf