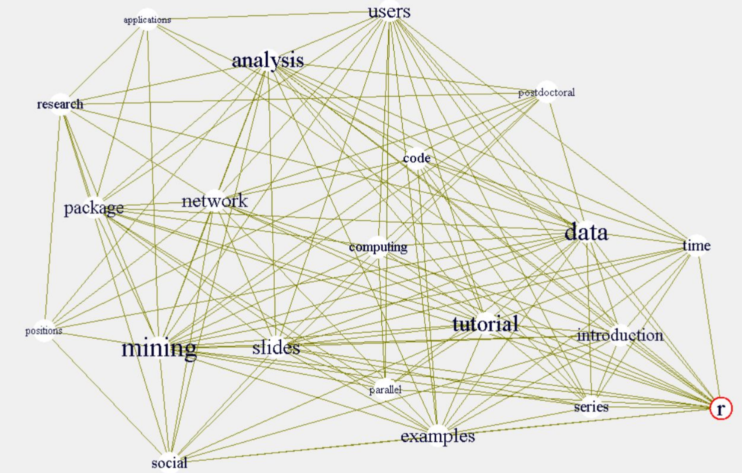
Data mining and network analysis IDN0110
Home assignment 3. Social Network Analysis
Report prepared by: Lizaveta Miasayedava (184361IVGM)
Implementation: *my_graph_measures.R, my_graph_helper_functions.R, test_graph_measures.R, test_graph_building.R*

Social networks analysis can be considered the analysis of graph structured data generated by web-based social applications that consist information about user preferences, their connections, and their influences on others, etc. A social network can be represented as a graph G = (N,A), where N is the set of n nodes and A is the set of m edges. Each individual (actor) in the social network can be represented by a node in N. The edges represent the connections (e.g. friendship links in Facebook) between the different actors. The nodes in N may have content associated with them (e.g. comments or documents posted by social network users).

For the social network analysis Twitter text data is used ("termDocMatrix.rdata", the Data webpage). The terms can be taken as people and the tweets as groups on LinkedIn, and the term-document matrix can then be taken as the group membership of people. The network of terms is built based on their co-occurrence in the same tweets, which is similar with a network of people based on their group memberships. At first, a term-document matrix, termDocMatrix, is loaded and then transformed into a term-term adjacency matrix, where the rows and columns represents terms, and every entry is the number of co-occurrences of two terms, based on which a graph is built.



The graph to shows the relationship between frequent terms.The label size of vertices is based on their degrees, the terms with higher degrees have larger labels.

Social networks have some common content and structural properties of node attributes that were observed in the analysis of social networks, such as:

Homophily: nodes that are connected to one another are more likely to have similar properties (individuals who are linked may often share common beliefs, backgrounds, education, hobbies, or interests).

Triadic closure (a tendency of networks to cluster, correlation of edges in social networks graphs): if two individuals in a social network have a friend in common, then it is more likely that they are either connected or will eventually become connected in the future.

The concept of **triadic closure** is related to the *clustering coefficient* of the network (a measure of the tendency of a network to cluster).

Clustering coefficients are estimated over each node i of an undirected network G=(N,A):

$\eta(i) = |\{(j, k) \in A : j \in Si, k \in Si\}| / C_n^2$; $\eta(i)$ - the clustering coefficient of the node i, Si - the set of nodes connected to i, n - the power of Si.

In other words, the clustering coefficient of the node i is the fraction of the number of existing edges between the nodes in Si over the number of all possible edges between nodes in Si.

The network *average clustering coefficient* is the average value of η(i) over all nodes in the network. The values of clustering coefficients range from 0 to 1. The higher clustering coefficient (closer to 1), the higher clustering tendency.

For example, for the given Twitter terms graph, its node “analysis” has 18 nodes (including the node itself) in the neighbourhood set S, the number of existing edges between these 18 nodes is 108, where the number of all possibles edges between 18 nodes is $C_{18}^2 = 153$, the clustering coefficient of the node “analysis” is 108/153=0,7.

Due to the property of **preferential attachment** of social networks**,** the likelihood of a node receiving new edges increases with its degree (highly connected individuals typically find it easier to make new connections). High-degree nodes tend to form **giant connected components** because newly incoming edges are more likely to attach themselves to the densely connected and high-degree nodes in the network. The presence of giant connected components affects the network clustering algorithms and may lead to unbalanced clusters. That’s why different measures are designed to be used in clustering algorithms.

Due to the property of preferential attachment, online networks have a typical structure having a small number of high-degree nodes (**hubs**) that attract most of the newly added nodes (they have ties to many actors and are in a position of better influence). Hubs can be considered central points of the network, they are usually connected to different regions of the network and affect different properties of the network (its density, connectivity, clustering behaviour, etc.).

In order to measure the centrality of the nodes there can be used the the *degree centrality.*

The degree centrality CD(i) of a node i of an undirected network is equal to the degree of the node Degree(i), divided by the maximum possible degree of the nodes (one less than the number of nodes n in the network):

CD(i) = Degree(i)/(n − 1)

For example, for the given Twitter terms graph, the degree of the node “analysis” is 17, the maximum possible degree of the nodes is 21-1=20, the degree centrality of the node “analysis” is 17/20=0,85.

In directed networks prestige measures are used:

Degree prestige PD(i) uses the indegree of the node. The idea is that only a high indegree contributes to the prestige because the indegree of a node can be viewed as a vote for the popularity of the node:

PD(i) = Indegree(i)/(n − 1)

The *gregariousness* GD(i) of a node i uses the outdegree of the node and defines the propensity of an individual to seek out new connections (such as following many other actors in Twitter), rather than his or her popularity with respect to other actors (as a prestige measure):

GD(i) = Outdegree(i)/(n − 1)

However, degree centrality and prestige measures don't consider nodes beyond the neighborhood of a given node i, the overall structure of the network is ignored to some extent. So some nodes can have high degree centrality, but can't be viewed as central because they are closer to the periphery of the network. To solve this problem closeness centrality and proximity prestige measures can be used.

The *closeness centrality* is defined for undirected networks. The closeness centrality CC(i) is the inverse of the average distance of other nodes to node i.

CC(i)=1/AvDist(i) ; $AvDist(i) = \frac{\sum_{j=1}^{n} Dist(i,j)}{n-1}$ - the average shortest path distance, starting from node i. *Dist(i, j)* - the shortest path distance between nodes i and j

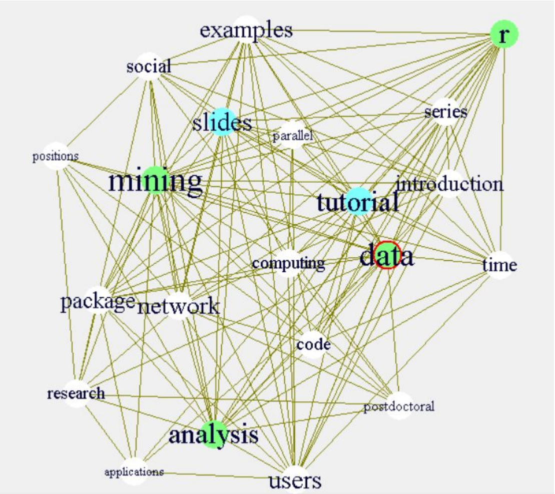
Nodes with the highest closeness centrality have the lowest average distance to other nodes.

The *proximity prestige* PP(i) can be used to measure prestige in directed networks:

PP(i) = InfluenceFraction(i)/AvDist(i); $AvDist(i) = \frac{\sum_{j \in Influence(i)} Dist(j,i)}{|Influence(i)|}$ - the average shortest path distance with respect to the set of nodes Influence(i) that can reach node i with a directed path. Distances are computed from node j to i, because a prestige measure, not a gregariousness measure is computed. *InfluenceFraction(i) = |Influence(i)|/(n-1)* - fractional size of the influence set of node i - a factor to penalize nodes that have less influence.

Higher values of proximity prestige indicate greater prestige. The highest possible proximity prestige value of 1 is realized at the central node of a perfectly star-structured network, with a single central actor and all other actors as its (in-linking) spokes.

For example, on the graph green nodes have highest degree centralities whereas blue nodes have highest closeness centralities.



Criticality (the number of shortest paths that pass through the node) defines actors that have the greatest control of the flow of information between other actors in a social network. In order to consider criticality there can be used the measure of *betweenness centrality* CB(i):

$$CB(i) = \sum_{j<k} fjk(i)/C_n^2$$

fjk(i) = qjk(i)/qjk - the fraction of pairs fjk(i) that pass through node i (indicates the level of control that node i has over nodes j and k in terms of regulating the flow of information between them). *qjk* - the number of shortest paths between nodes j and k; *qjk(i)* - the number of these pairs that pass through node i. The betweenness centrality lies between 0 and 1, higher values indicate better betweenness.

Another problem that can be addressed in the context of social network analysis is link prediction - the prediction of future links between pairs of nodes in the network using content (homophily property: nodes that have similar content are more likely to become linked) or structural similarity (triadic closure: 2 nodes that share similar nodes in their neighborhood are more likely to be connected in the future, if they are not connected yet).

The examples of structural measures for link prediction:

Common neighbour measure between node i and j i equal to the number of common neighbours between nodes i and j:

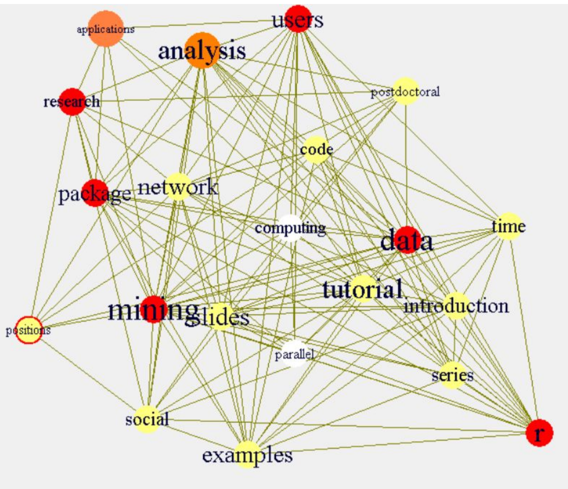
$CommonNeighbours(i,j) = |Si \cap Sj|$; Si - the neighbour set of node i; Sj - the neighbour set of node j.

Jaccard Measure between node i and j is equal to the Jaccard coefficient between their neighbour sets Si and Sj (the fraction of common nodes over all the neighbour nodes of the nodes i and j).

$$JaccardMeasure(i,j) = \frac{|Si \cap Sj|}{|Si \cap Sj|}$$

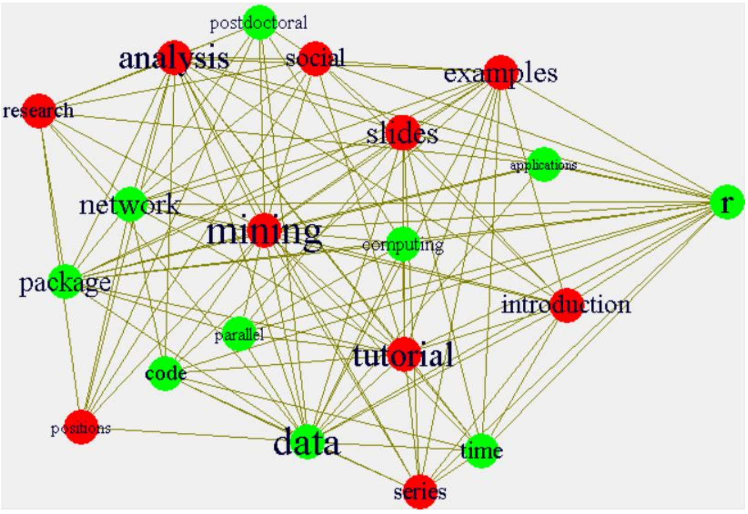
For example, for the nodes “applications” and “analysis” yellow nodes represent all the neighbourhood nodes of these 2 nodes (the join of Si and Sj) , red nodes - common nodes (the intersection of Si and Sj).

So, the common neighbour measure for the nodes “applications” and “analysis” is 6 (the number of common nodes). And the Jaccard measure is 6/17 (the number of red nodes/the number of yellow nodes).



Community detection

The problem of network clustering, or community detection, can be described as the partitioning of the network into k sets of nodes, such that the sum of the weights of the edges with vertices in different partitions is minimized. For the implementation of clustering the Kernighan–Lin Algorithm was chosen.



The results of the Twitter text graph partitioning into 2 subsets with the implemented Kernighan–Lin Algorithm

The Kernighan–Lin algorithm partitions graph into 2 equal subsets. It starts with 2 equal randomly partitioned subsets of nodes and then iteratively improves this partitioning until convergence. The iterative improvement is performed by determining sequences of exchanges of nodes between partitions that improve the clustering objective function as much as possible.

Implementation: *my_graph_clustering.R*