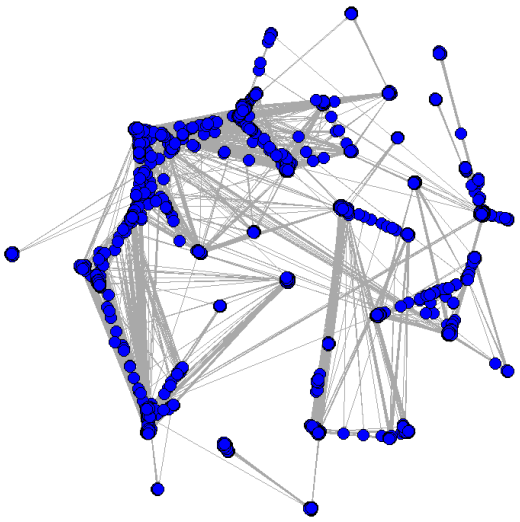
1. Introduction

This exercise focuses on two crucial aspects of social network analysis. Through exploring the several available measures of network centrality, we aim to identify the most integral nodes within a Facebook friend network. Once this is established, we also run community detection algorithms to determine what sub-communities exist within the overall network.

2. Data selection

Our selected data is a Facebook friend network dataset courtesy of Jure Leskovec at Stanford University (2012). A summary of the dataset statistics and initial network plot can be found below:

Dataset statistics	
Nodes	4039
Edges	88234
Nodes in largest WCC	4039 (1.000)
Edges in largest WCC	88234 (1.000)
Nodes in largest SCC	4039 (1.000)
Edges in largest SCC	88234 (1.000)
Average clustering coefficient	0.6055
Number of triangles	1612010
Fraction of closed triangles	0.2647
Diameter (longest shortest path)	8
90-percentile effective diameter	4.7



3. Main model and analysis

To begin, we examined the following centrality measures on the dataset: degree, closeness, eigenvector, betweenness and Bonacich alpha. We will briefly explain each measure in the context of our dataset below:

- **Degree:** This is defined as the number of links a certain node has with other nodes. So, in this case, degree is the number of friends each Facebook user has. The idea is that the more connections (higher degree) a node has, the more influence they should have on the network.
- **Closeness:** This is an approach that measures the how long a flow takes to pass from a given node to all other nodes sequentially. In this case, closeness is the shortest path between one Facebook user to all other users in the network. The idea is that the quicker (more easily) a node can reach other nodes in the network, the more significance it has.
- **Eigenvector:** In this measure, relative scores are assigned to each node based on valuing connections to high-scoring nodes contributing more to a given nodes

score than equal connections to low-scoring nodes. In our case, the higher Facebook users' score, the higher its connections score as well.

- **Betweenness:** In this measure, one is evaluating a nodes role as a connector or bridge between other nodes in the network. In a given network, you determine the number of shortest paths existing between two nodes and note how many times those shortest paths pass through node *x*, determining the betweenness for node *x*. In the context of our model, betweenness is defining how often a Facebook user is the common connection (mutual friend) between two other users. A node with high betweenness thus has a great influence on the flow through the network.
- **Bonacich alpha:** An extension of eigenvector centrality, this measure has an additional aspect, being that node scores are determined with importance from external sources. For our model, this approach is considering both neighboring Facebook users as well as our own weighting of the importance of those neighbors.

Similarly, we chose specific community detection methods including: walktrap, edge betweenness, and fast and greedy:

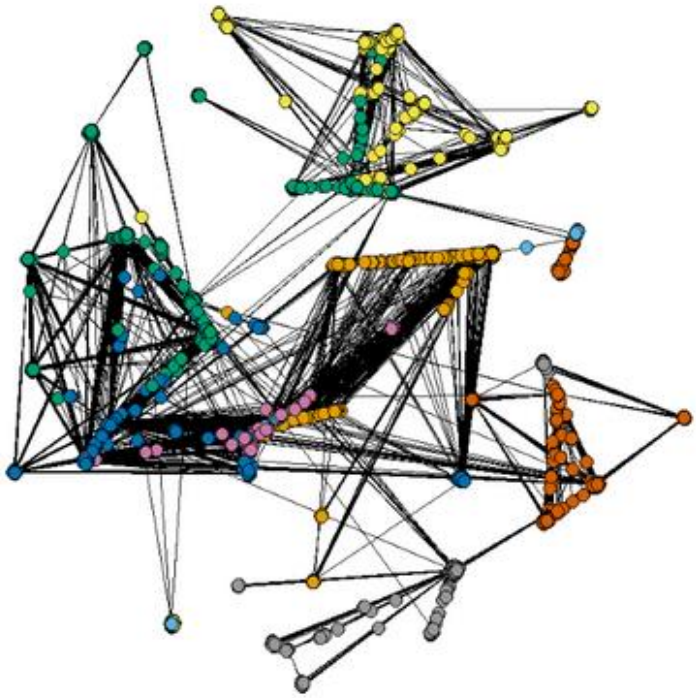
- **Edge Betweenness:** This measure quantifies the number of shortest paths which traverse a given node and each of which is associated with its own unique betweenness value. We pay specific attention to the nodes where we observe a higher betweenness score as these are the nodes which act as "bridges" between clusters of a network and their negation from the network would lead to a decrease in transfer/flow within the network.
- **Walktrap:** A hierarchical clustering algorithm which goes from bottom to top and works with the logic that the short-distance random walks remain within a given community, this algorithm starts with a completely non-partitioned network and then computes the distances between all adjacent nodes which lead to the identification of two distinct communities and a subsequent merge is performed which also updates the distances. This step is iterated (N – 1) times to completely run through the network and identify as many communities as possible. This algorithm is often used as a first approximation to get a feel for the network in question.
- **Fast and Greedy:** Another hierarchical approach which traverses bottom to top rather than top to bottom, optimizes the modularity in a "greedy" fashion. At the time of implementation, every node is part of a different community and with every iteration the communities get merged in a manner that the value of modularity is augmented to the highest level, the result is a grouping alongside a dendrogram. Given its uniqueness of having no parameters to fine tune and being relatively fast, this is one of the most useful approaches deployed as first approximation. The only shortcoming here is that for communities of a certain size, it would merge these with neighboring clusters/communities.

We then compared models in R to try and predict which nodes belong to which communities based on the explained centrality measures above, the results of which are discussed in the next section.

4. Discussion, conclusion and interpretation of main insights

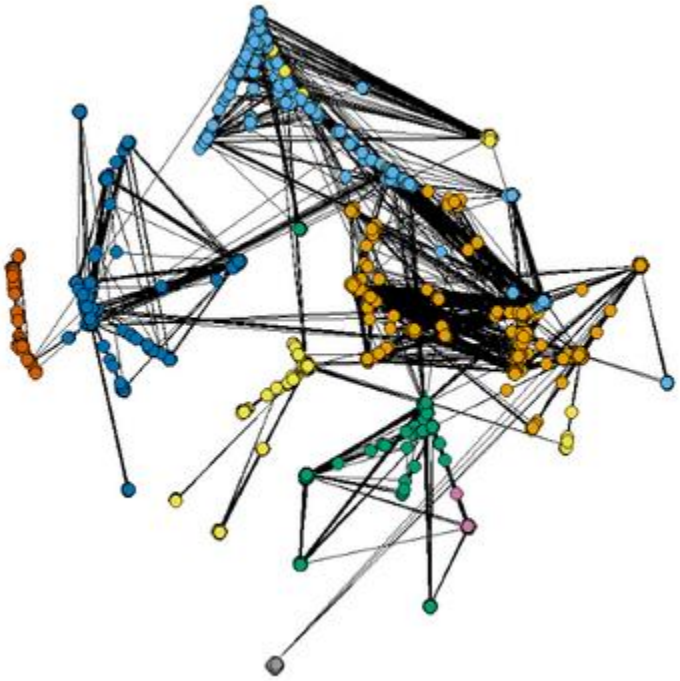
The approaches we tried output very different results, especially in number of communities detected. The Walktrap method finds 28 different user groups, though 10 of them are composed of less than 10 observations. Although it makes sense on a community detection perspective (a node that is too far away from any cluster to possibly belong to any existent ones), this level of detail will ensure the failure of any traditional classification algorithm to recognize these under represented clusters. This algorithm should be preferred if we are interested in identifying small outlier communities and nodes individually.

Visualization of Walktrap communities



The Fast & Greedy approach addresses this problem in the sense that it tries to regroup nodes together to its maximum up to the point at which the modularity doesn't increase anymore. In practice, it means small clusters will tend to get 'absorbed' by bigger ones, resulting in this case in 13 balanced communities. This approach is useful to segment a population in a broader way and serves our purpose of creating a balanced dataset in which each community is fairly represented.

Visualization of Fast & Greedy communities



In a final step we split the data into a train and test set in order to predict based on the centrality measures to which community one node belongs. This was done for communities detected with the walktrap as well as for the fast and greedy by using the following algorithms:

K-Nearest Neighbors

This algorithm classifies nodes into communities based on the nodes that are closest to it. Closeness is in our case defined by the centrality measures. We chose this algorithm as we assumed that nodes with similar values in the centrality measures belong to the same community.

Random Forest

A great, flexible and easy to deploy machine learning algorithm which produces solid results even without hyper-parameter tuning is the Random Forest Model. Due to its simplicity and its flexibility to be used for multiple cases like classification and regression it is one of the go-to algorithms.

A supervised algorithm which builds multiple decision trees and binds them to get an accurate and constant prediction. Most of the times, the algorithm is trained with the “bagging” ideology. The algorithm introduces randomness to the model as it increases the number of trees and checks for the best feature amongst the randomly distributed features instead of looking for a node to split. This allows for diversity within a model and leads to better results yielded by the model.

Extreme Gradient Boosting

As the Random Forest delivered good results, we decided on using a second tree-based approach. Extreme Gradient Boosting (xgboost) is an algorithm that creates small trees with only a few splits sequentially and learns from the errors made by the previous trees.

Support Vector Machine

An SVM is a discriminative algorithm which defines the data using a separating hyperplane. Using the training data, which is properly labelled, the algorithm works around to produce an optimally placed separating hyperplane which would categorize all future examples. In a 2D model, this hyperplane is a line which neatly classifies classes on each side of it.

Modeling Results

Below is a table presenting the accuracy scores of 4 models built with both Fast & Greedy and Walktrap communities as a target variable.

	Fast & Greedy	**Walktrap**
KNN	0.469	0.322
Random Forest	0.942	0.597
XGBoost	0.903	0.560
SVM (linear)	0.469	0.265

There is a clear difference in results between predicting communities assigned with Walktrap versus the ones assigned with the Fast & Greedy algorithm. This can partially be explained by the difference in class distribution, and the number of communities itself. Another explanation is that the Walktrap process uses randomness as the main component of its community identification, while F&G relies on the computation of modularity. The intuition is that modularity is closely correlated to centrality measures, which allows algorithms such as Random Forest to find meaningful information in the independent variable to predict communities.

A last discovery is the variable importance, which is consistently ranked in the same order. Eigenvector is the prime source of information for community identification, followed by the Bonacich alpha, closeness, degree and finally betweenness. As a final summary one can conclude that by using the centrality measures explained above and the right algorithm one is able to predict the community with a decent accuracy.