Collaboration Network Analysis

R04921049 柯劭珩 B01901121 李律慈

Abstraction

We investigate the academic collaboration behavior between students in this project. A social network is created based on collaboration specifications in the class homeworks, and the amount of collaboration work is aggregated and utilized as edge weight. We employ *Python - networkx* to calculate important measures in the network, such as degree, eigen-centrality, page-rank and hub/authorities of the HITS algorithm. These attributes are used by *Cytoscape*, in order to visualize the network. They are also mined in classification and association algorithms, such as *J48* decision tree and *Apriori* association.

Introduction

In economics and network science, there are lots of work about measuring the importance of an arbitrary node in a given network, from various centrality definitions to more complicated algorithms like HITS. They all make sense in special cases, but when facing a new network, we are not completely sure about which gives the most information.

On the other hand, there are more and more data mining in the field of MOOC(massive online open course). On the leading platforms, such as Coursera, data of millions of users are mined and investigated for various purposes. As recent as 2014, it was shown that one can predict which student will drop out from the course, at a decent rate, by only mining their actions: logging in, watching videos, replying and posting in the forums.

However, the above method views the users as individuals and omit their interactions altogether. Not taking that into account, we may be losing a huge part of information available. In this project, we try to combine the network analytics and data mining, by using the measurements of network in mining. Although the data we utilize lacks volume, it is clearly shown that some of the network measurements can be used in classification; that is, predicting the performances of students.

Data Collection

The data comes from the Algorithms class in NTUEE, offered by Prof. Ho-Lin Chen. In this course, there are 25 homework problems throughout the semester, and all the students are required to finish them individually. However, they can collaborate with peers and specify such collaboration in the homework. The raw data contains all the collaboration specifications, in the form of "Student A -> Student B in problem X."

A social network G(V,E) is then created, with V being the set of all students (152 in total), and an edge from student A to student B is created given the example above. All the edges (directed, 866 in total) are then weighted by the proportion in grades of the corresponding problem. Finally, duplicated edges are combined into one, with the weight aggregated. We obtain some other attributes for every student, which is listed below.

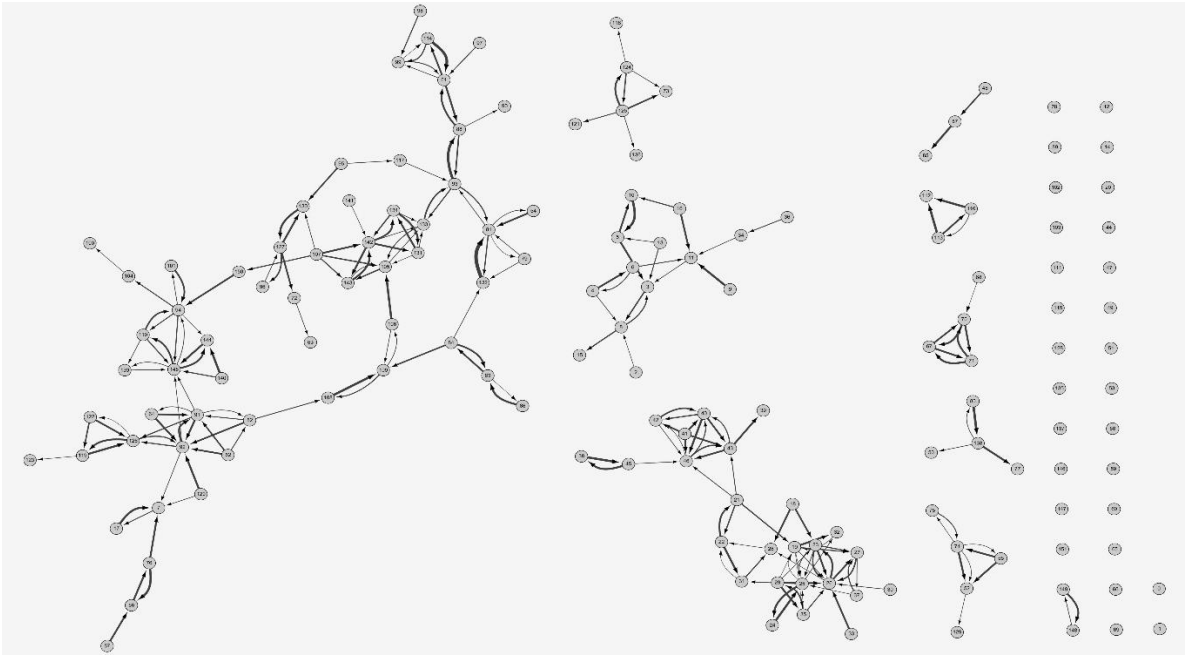
Attribute Name	Type	Values	Usage/Meaning
ID	nominal	1~152	linking and binding data
identity	nominal	B(under-graduated), R(graduated), O(others)	classification
Homework	numeric	0~107	HW grade
Midterm	numeric	0~100	Midterm grade
Final	numeric	0~85	Final grade

The social network was fed into the *Python-networkx* library as input, as listed below:

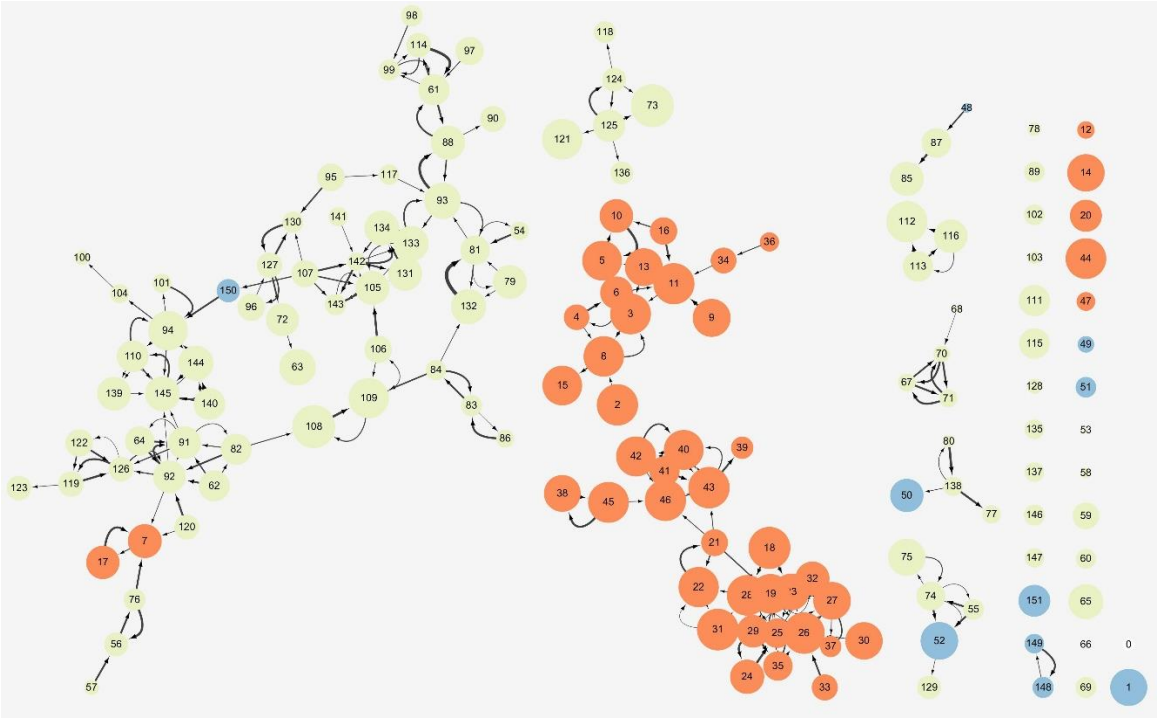
Attribute Name	Type	Function used	Attribute Name	Type	Function used
Indegree	numeric	in_degree_centrality	Page rank	numeric	pagerank
Outdegree	numeric	out_degree_centrality	hub	numeric	hits
Eigenvector Centrality[1]	numeric	eigenvector_centrality	authorities	numeric	hits

Visualization

Before the data is mined, a series of visualization is performed to help find meaningful trends in data. We used *Cytoscape*, a free network-visualizing software. It also has the ability to calculate and demonstrate some basic graph parameters for smaller networks.

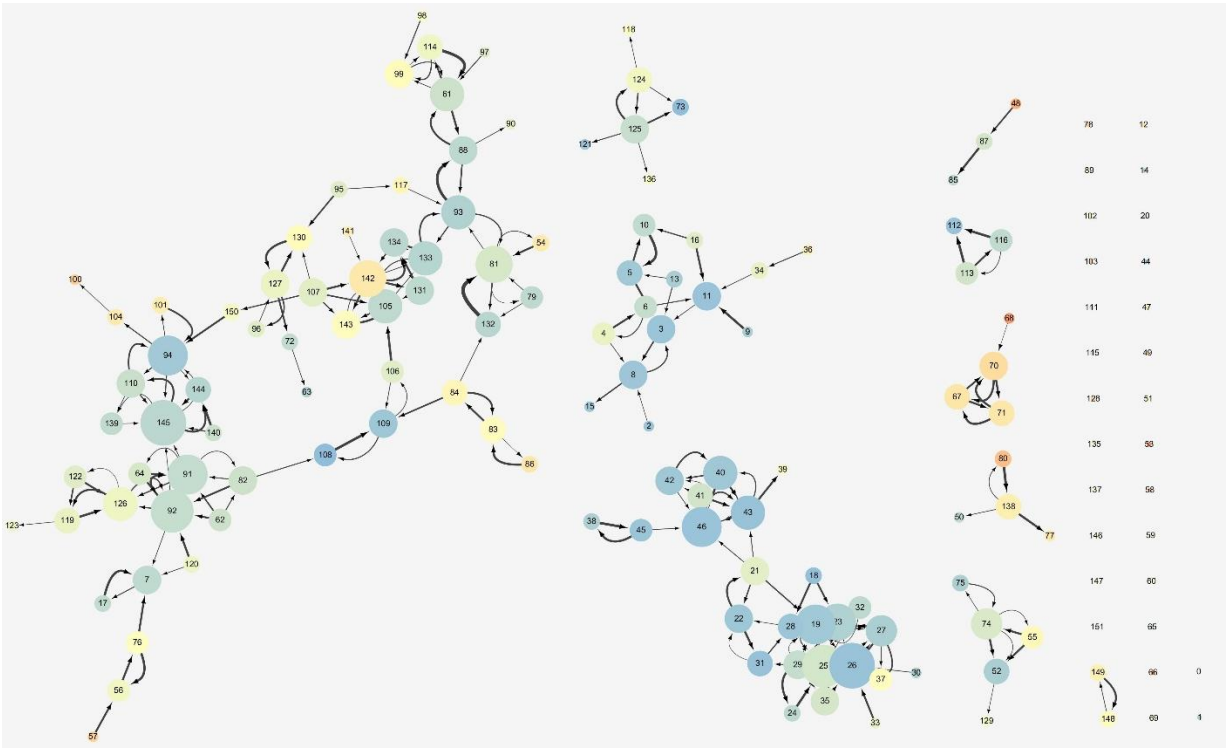


The above figure shows the network. Each small circle represents a node, with its ID inside the circle. We can easily recognize connected components, as there are 10, omitting singleton nodes. The largest connected component has 58 nodes, more than a third of the network. The edges, directed and weighted, are shown in arcs of different widths. The width is proportional to the edge weight. We use the Prefused Force Directed Layout[2] to demonstrate the network.



We added more features to the network. In the above figure, the color represents the background of students. Graduated students are shown in light green nodes, and under-graduated students are shown in brown nodes. The rest, in blue, are other types of students, like transferring students. The separation between bunches of nodes is very clear, as the interaction between graduated and under-graduated students being nearly nonexistent. The exceptions, nodes 7 and 17, are under-graduated student in the graduated component, but they don't play important roles there, either.

The node size are proportional to their Homework grades. We can observe that bigger nodes tend to appear in the center of a component; intuitively, such students are the target of collaboration actions. The smaller nodes, with weaker grades (and maybe ability), lack incoming actions. Finally, we can see that brown nodes are in average larger than green nodes!

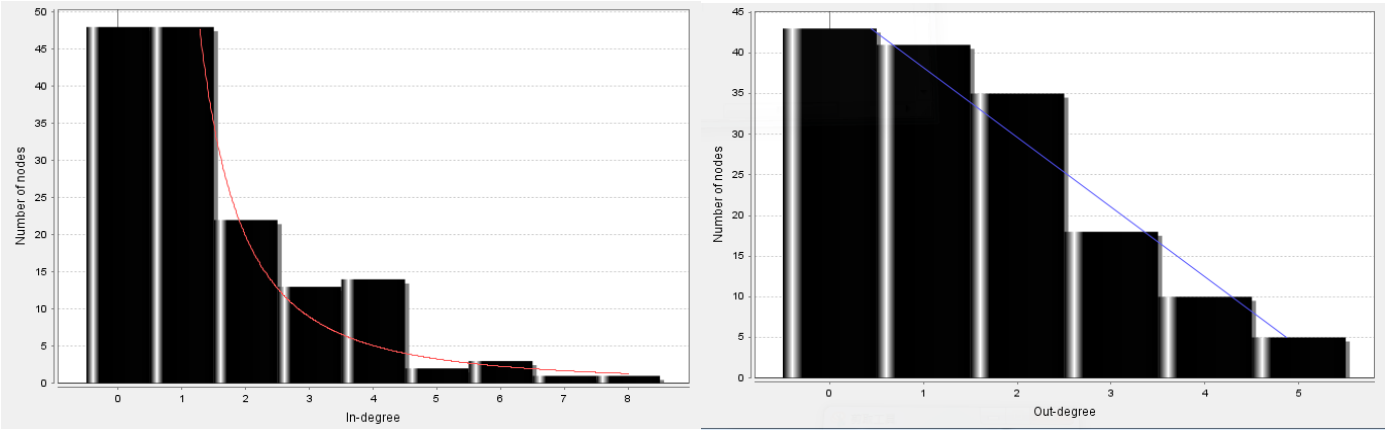


The last observation can be verified in a different setting of visualization. In the above figure, the color of the node represents the homework grade. The darker nodes are those with better grades, and the lighter nodes have weaker grades. It is again observed that the giant component has weaker grades. Also, the node size is now proportional to its total degree. We can now observe another intuitive fact: the nodes with high degrees tend to have better grades. The isolated nodes, lacking any collaboration, has zero degree and thus zero area.

Network Analyzing

Our collaboration network is sparse. There are 152 nodes (28 isolated) and only 252 weighted edges, which leads to a 0.022 network density. However the global clustering coefficient is as large as 0.174, qualifying as a decent clustering phenomenon. The local global clustering coefficients float around 0.3.

We can look at the in-degree and out-degree distributions, shown in the figure below. The distributions differ obviously. The in-degree distribution is more similar to a power-law distribution, the distribution in small-world or scale-free networks. However the out-degree distribution is more linear, which can be intuitively understood since that if a student has difficulty on his homework, he only needs one reliable source to collaborate and solve the problem. Once he finds such reliable source he will likely go to it afterwards throughout the semester. No students asked more than five peers, as shown in the out-degree figure.



Classification

We use *Weka* as the mining tool. We are interested in predicting the performance of a student, given the whole picture of the network and the role he plays in it. To filter out noises and simplify the results, we discretized the scores into three categories: {High, Medium, Low} in relative sense. Hence, the top third goes to High, and the medium third goes to Medium, etc. This serves as classes to determine in a classification problem.

We tried to discretize other parameters as well. Since some of the parameters take a wide span of values (Eigen-centrality is from one tenths to like 1E-50), we duplicated a version of data with all numeric data values discretized into {High, Medium, Low} in relative sense.

We then run classification algorithms on the data. Below is part of one of the results of running the J48 decision tree on the numerical (non-discretized) data. This decision tree can classify 80% of the instances. Since our data lacks volume, we simply use the whole dataset in training, instead of keeping a part for checking. The data is so small that k-fold cross validation doesn't give any meaningful results.

The interesting part is, the first attribute that comes into play is *Student_Class*, or identity! A student's identity info, that he is graduated or not, gives the most information in predicting the grade. This is an unexpected result. However, network parameters do play some roles; we can see the hub parameter in HITS algorithm, the in/out degrees in the decision tree.

```
Student_Class = R
|   hits_hub <= 0.00009
|   |   in_degree(aggregated) <= 275
|   |   |   in_degree(unaggregated) <= 0.020979
|   |   |   |   out_degree(unaggregated) <= 0.006993
|   |   |   |   |   out_degree(unaggregated) <= 0
|   |   |   |   |   |   in_degree(aggregated) <= 15: Medium (3.0/1.0)
|   |   |   |   |   |   in_degree(aggregated) > 15: High (4.0/1.0)
|   |   |   |   |   out_degree(unaggregated) > 0
|   |   |   |   |   |   in_degree(unaggregated) <= 0.013986
|   |   |   |   |   |   |   in_degree(aggregated) <= 35: Low (11.0/2.0)
|   |   |   |   |   |   |   in_degree(aggregated) > 35: Medium (5.0)
|   |   |   |   |   |   in_degree(unaggregated) > 0.013986: High (3.0/1.0)
|   |   |   |   out_degree(unaggregated) > 0.006993
|   |   |   |   |   in_degree(aggregated) <= 160
|   |   |   |   |   |   in_degree(unaggregated) <= 0.006993
|   |   |   |   |   |   |   page_rank <= 0.00296: Medium (12.0/2.0)
|   |   |   |   |   |   |   page_rank > 0.00296: Low (4.0)
|   |   |   |   |   |   in_degree(unaggregated) > 0.006993: Medium (9.0/1.0)
|   |   |   |   |   in_degree(aggregated) > 160: Low (5.0)
|   |   |   in_degree(unaggregated) > 0.020979: High (4.0/2.0)
|   |   in_degree(aggregated) > 275
|   |   |   in_degree(unaggregated) <= 0.020979: High (4.0)
|   |   |   in_degree(unaggregated) > 0.020979: Medium (4.0/1.0)
|   hits_hub > 0.00009: High (9.0/3.0)
Student_Class = B
|   in_degree(aggregated) <= 125
```

We also tried to delete the identity information before running classification algorithms. The following figure shows part of the result of running on the numerical dataset excluding *Student_Class*. The prediction rate is 83%, and we can see eigenvalue-centrality, pagerank value and authorities all appear in the tree.

```
page_rank <= 0.007851
|   out_degree(unaggregated) <= 0.013986
|   |   out_degree(unaggregated) <= 0
|   |   |   page_rank <= 0.005308
|   |   |   |   in_degree(aggregated) <= 20: Medium (4.0/1.0)
|   |   |   |   in_degree(aggregated) > 20: High (3.0)
|   |   |   page_rank > 0.005308: Low (3.0/1.0)
|   |   out_degree(unaggregated) > 0
|   |   |   out_degree(aggregated) <= 15
|   |   |   |   in_degree(unaggregated) <= 0.006993: Low (5.0)
|   |   |   |   in_degree(unaggregated) > 0.006993: High (3.0/1.0)
|   |   |   out_degree(aggregated) > 15
|   |   |   |   out_degree(unaggregated) <= 0.006993
|   |   |   |   |   in_degree(aggregated) <= 20: Low (12.0/5.0)
|   |   |   |   |   in_degree(aggregated) > 20: Medium (9.0/1.0)
|   |   |   |   out_degree(unaggregated) > 0.006993
|   |   |   |   |   degree(aggregated) <= 60: High (3.0)
|   |   |   |   |   degree(aggregated) > 60
|   |   |   |   |   |   in_degree(aggregated) <= 110
|   |   |   |   |   |   |   degree(aggregated) <= 105: Low (2.0)
|   |   |   |   |   |   |   degree(aggregated) > 105: Medium (11.0/
|   |   |   |   |   |   in_degree(aggregated) > 110: High (3.0/1.0)
|   out_degree(unaggregated) > 0.013986: Medium (16.0/3.0)
page_rank > 0.007851
|   eigen_centrality <= 0.000222
|   |   in_degree(unaggregated) <= 0.006993
```

From the two settings we have already seen all of our calculated parameters come into play in the tree. The prediction rate is lower, around 70% if we use discretized values, as we give an example below. From the result we can see that discretizing doesn't lose too much ground in classification.

```
page_rank_class = High
|   eigen_class = Medium
|   |   in_degree_unagg_class = High: High (9.0)
|   |   in_degree_unagg_class = Medium
|   |   |   out_deg_agg_class = Low: High (2.0)
|   |   |   out_deg_agg_class = Medium: High (0.0)
|   |   |   out_deg_agg_class = High: Medium (2.0)
|   |   in_degree_unagg_class = Low: High (0.0)
|   eigen_class = High: High (27.0/13.0)
|   eigen_class = Low: Low (2.0)
page_rank_class = Medium
|   out_degree_unagg_class = Low
|   |   deg_agg_class = Medium: High (4.0)
|   |   deg_agg_class = High
|   |   |   hits_auth_class = None: High (0.0)
|   |   |   hits_auth_class = Low: Medium (3.0/1.0)
|   |   |   hits_auth_class = Medium: High (3.0)
|   |   |   hits_auth_class = High: High (0.0)
|   |   deg_agg_class = Low: Medium (11.0/6.0)
|   out_degree_unagg_class = Medium
|   |   eigen_class = Medium: Low (6.0/2.0)
|   |   eigen_class = High: High (5.0/1.0)
|   |   eigen_class = Low: Medium (2.0/1.0)
|   out_degree_unagg_class = High: Medium (6.0/2.0)
page_rank_class = Low
|   in_deg_agg_class = High: Medium (0.0)
|   in_deg_agg_class = Low
|   |   eigen_class = Medium: Low (4.0/1.0)
|   |   eigen_class = High: Medium (3.0)
|   |   eigen_class = Low
|   |   |   deg_agg_class = Medium: Medium (6.0/1.0)
|   |   |   deg_agg_class = High: Medium (1.0)
|   |   |   deg_agg_class = Low
|   |   |   |   hits_hub_class = None
```

Since the identity information comes out of nowhere to appear important, we take one step further, separate the dataset by *Student_Class*. As a result, we have one instance set that includes only graduated students, and one that has only under-graduated students. Since there are only 11 Other students, we did not make the third dataset. Below is an example of decision tree on the graduated students dataset. The prediction rate sits at 81%.

```
hits_hub <= 0.00009
|   in_degree(aggregated) <= 275
|   |   in_degree(unaggregated) <= 0.020979
|   |   |   out_degree(unaggregated) <= 0.006993
|   |   |   |   out_degree(unaggregated) <= 0
|   |   |   |   |   in_degree(aggregated) <= 15: Medium (3.0/1.0)
|   |   |   |   |   in_degree(aggregated) > 15: High (4.0/1.0)
|   |   |   |   out_degree(unaggregated) > 0
|   |   |   |   |   in_degree(unaggregated) <= 0.013986
|   |   |   |   |   |   in_degree(aggregated) <= 35: Low (11.0/2.0)
|   |   |   |   |   |   in_degree(aggregated) > 35: Medium (5.0)
|   |   |   |   |   in_degree(unaggregated) > 0.013986: High (3.0/1.0)
|   |   |   out_degree(unaggregated) > 0.006993
|   |   |   |   in_degree(aggregated) <= 160
|   |   |   |   |   in_degree(unaggregated) <= 0.006993
|   |   |   |   |   |   page_rank <= 0.00296: Medium (12.0/2.0)
|   |   |   |   |   |   page_rank > 0.00296: Low (4.0)
|   |   |   |   |   in_degree(unaggregated) > 0.006993: Medium (9.0/1.0)
|   |   |   |   in_degree(aggregated) > 160: Low (5.0)
|   |   in_degree(unaggregated) > 0.020979: High (4.0/2.0)
|   in_degree(aggregated) > 275
|   |   in_degree(unaggregated) <= 0.020979: High (4.0)
|   |   in_degree(unaggregated) > 0.020979: Medium (4.0/1.0)
hits_hub > 0.00009: High (9.0/3.0)
```

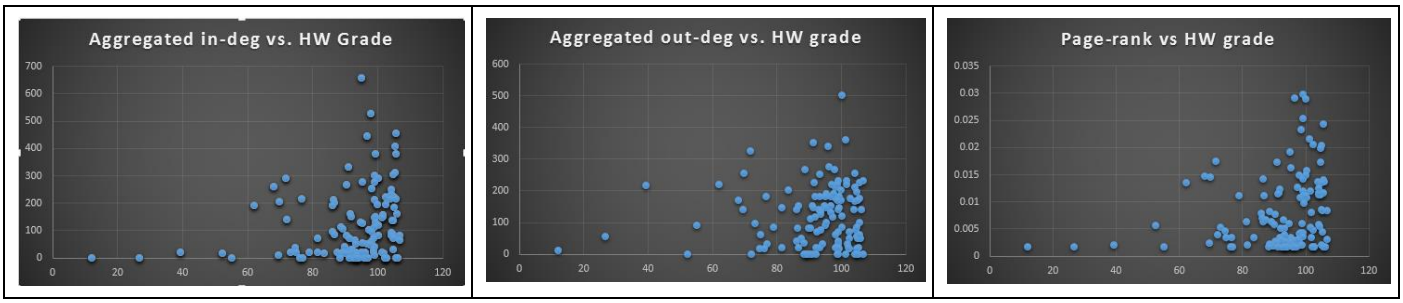
So, what do we found in classification? Apart from that the identity, or background, says much about a student's grade, we also find that the other parameters can really predict the grade as well, even when we scratch the identity away. It does not appear like that any of the parameters stand out from the picture, as they take turns to appear on higher levels of the trees; but in and out degrees, which are the most fundamental measures, seem to have an edge.

Association

We also tried the *Apriori* association algorithm on the discretized dataset, in order to find hidden trends in our data. Lots of good rules were found. However, none of them says anything about grade, what we care. All the good rules point to the very, very strong relation between the network parameters: `In_Degree = HIGH => Authority = HIGH` states something we are very sure already. Such relation hid anything related to the grade.

Observation

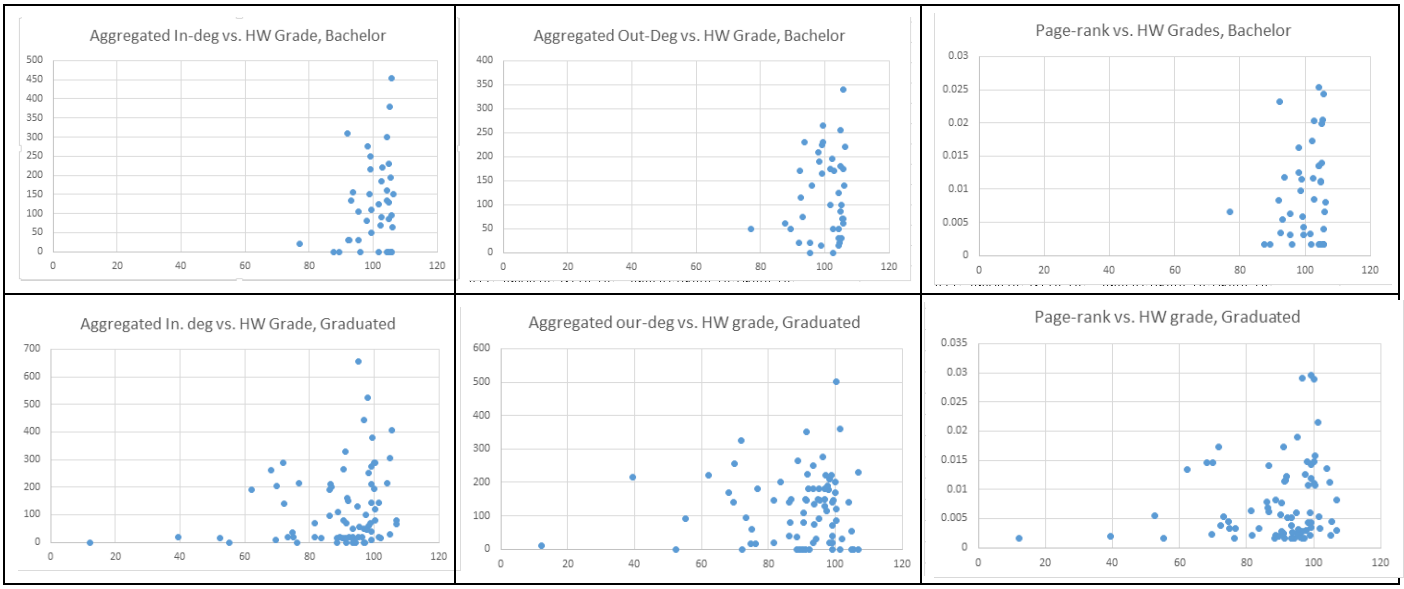
Our naïve expectation was that high collaboration will be highly related to high grades. Our data mining results weren't that pretty, and the unexpected *Student_Class* jumped out. However, we still see some extent of relation, so we plotted some figures for deeper inspection:



When we plot our network parameters with grades, a clear pattern emerges. There are triangles in virtually every figure, with those of in/out degree and page-rank being the more obvious ones. If there exists a simple relation between the axes, it will be a straight line; a triangle indicates a single-sided relation instead. It seems that though high grade does not imply high collaboration, but the opposite direction of this relation holds: high collaboration brings high grade. There are almost no data points in the upper-left part, which suggests high collaboration and low grade.

This observation holds after we separate groups, as we show below. The triangular pattern appears in separated datasets. The triangle is flatter in the graduated student dataset, which suggests high variety. The grades for under-graduated student are higher, in comparison.

This one-sided relation is already intuitive, and doesn't need data mining to verify; we need only statistical methods to verify it. But its one-sided nature blurs the whole picture and deters the classification results, since the relation is not simple and straight. This at least partially explains the lack of quality of our data mining results.



Conclusion

In this Algorithms class, after turning the whole class into a social network, we found that several network parameters are related to homework grades. From classification algorithms, and from simple observation as well we also found that identity(background) plays a big role in grade, at least in this course. An one-sided relation is found between the grade and any one of our network parameters, especially in degree, out degree and page-rank.

Drawback

The data lacks volume. Though it didn't lead to very poor result of classification, cross-validation became impossible. Since the network itself is too sparse, current network predicting method cannot be employed on this work. However, we now have some extent of confidence that network parameters are useful in such collaboration network, and are ready to gather larger data sets for deeper inspection.

Appendix - Eigenvalue Centrality

In graph theory and network analysis, indicators of centrality identify the most important vertices within a graph. Eigenvector centrality is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the nextwork based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes. For a given graph G:=(V,E) with |V| number of vertices let A = $(a_{v,t})$ be the adjacency matrix. The centrality score of vertex v can be defined as:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} a_{v,t} x_t$$

In general, there will be many different eigenvalues for which an eigenvector exists. However, the additional requirement that all the entries in the eigenvector be positive implies that only the greatest eigen value results in the desired centrality measure. The vth component of the realted eigenvector then gives the centrality score of the vertex v in the network.

Appendix – Prefused Force Directed Layout

The force-directed layout is a layout based on the "force-directed" paradigm. This layout is based on the algorithm implemented as part of the prefuse toolkit provided by Jeff Heer.

This layout method positions graph elements based on a physics simulation of interacting forces; by default, nodes repel each other, edges act as springs, and drag forces (similar to air resistance) are applied. This algorithm can be run for multiple iterations for a run-once layout computation or repeatedly run in an animated fashion for a dynamic and interactive layout.

The running time of this layout algorithm is the greater of O(N log N) and O(E), where N is the number of nodes and E the number of edges. The addition of custom force calculation modules may, however, increase this value.

Suggestions to Course

The quality of exercises varies a lot. Overall, the R(swirl) courses were fun despite some technical problems. But the Python homework are virtually just examples in the documents. Those homework is hence too easy and doesn't require fully understanding the contents. On the other hand, the Spark/Hadoop part is way too hard; it even requires additional resources, which is troublesome.

Among the hours put in, we can say that over 50% time was used in setting environments. Only in the R(swirl) courses we were brainstorming, interacting with the homework. There are 8 assignments (7 HW plus the final), and two-thirds of them are in the second half. Maybe it can be balanced a bit.