# Rating Modules on cnx.org: Using Network Structure as a Feature in Regression

#### Austen Head

December 10, 2010

#### Abstract

Machine learning methods are applied to features scraped from the website cnx.org to predict page rating. In addition to the explicit features that are scraped from the pages on this website, there is an implicit network in these pages (through internet hyperlinks, key words, authors, etc.). This paper explores the predictive power that is gained by taking advantage of some aspects of this underlying network.

## 1 Introduction

The website cnx.org hosts scholarly content. Users create "modules" which each contain a chunk of information on a particular topic. Users can also create "collections" which are sets of modules that are all related in some way. Each module and collection has a single author who can invite other cnx.org members to be maintainers of the page (on most pages, the only maintainer is the author). Only the author and maintainers of a module are able to edit that module. Only the author and maintainers of a collection are able to choose which modules are part of that collection, but cnx.org encourages users to take other users' modules for their own collections. For example, the module "Galileo's Telescope" is used in three collections ("solar system", "Galileo Project", "History of Western Art and Civilization"). The collection "solar system" has nine modules in it.

On November 8, 2010, cnx.org hosted 17,189 modules and 1,030 collections. Many modules are not in any collection, and some are in multiple collections. The quality of the modules and collections is quite variable. It is often difficult to find modules or collections that are accurate and informative. Although cnx.org has a rating system for users to rate modules, there is no rating system for collections and modules can only be rated with 1 to 5 stars by users who are logged in. Each module has an "average rating" score from 1 to 5 by steps of 1/2 or is "Not yet rated" (the vast majority falls into this latter category). Very few modules have ratings (0.19% 1 star, 0.10% 2 stars, 0.27% 3 stars, 0.02% 3.5 stars, 0.52% 4 stars, 0.07% 4.5 stars, 1.98% 5 stars, 96.84% unrated). And those that do are mostly rated as 5 stars (62.8%).

I will predict how good each module is (based on module ratings) based on the following features from the pages. (i) # ratings, (i) views: total, (i) views: 10/31/10 to 11/7/10, (o) is it a module or collection, (f,n) language it's written in, (f,n) subject, (n,i) lenses it's a part of, (n,i) collections it is a part of, (n,i) key words, (n,i) tags, (n,i) related modules, (n) links to other mods or cols, (i) # links (any website), (i) # links (any cnx.org page), (i) # images on the page, (n,i) collection contributors, (n) author, (n,i) maintainers, (i) revision dates, (o) title, (o) summary of article. For additional information on what these features mean, see Figure 1. Items marked with an (i) are integer values, (f) are factor valued (unordered with less than 100 levels), (n) are network valued either explicitly as in the case of "related modules" or implicitly as being two hops away on a bipartite network such as two modules sharing a common "collections it is a part of" or a common "key words", (o) are features that I omitted in my analysis. The feature "is it a module or collection" was omitted because there are no collections with any ratings which makes the collection pages only useful through organizing the modules in network structures. The "title" and "summary of article" were difficult to work with since there are many pages in foreign languages and there are many pages which did not have a summary which made it impractical to try to use any sort of naive Bayes methods that are useful in problems such as spam classification. The features which are labelled both (n,i) have a network structure but counted the number of occurrences for example in "tags" I created the bipartite network structure and also counted the number of tags each module had (this is just the degree of each module in that bipartite graph). Similarly (f,n) were used both as factors. There are 50 languages including British English and American English as separate languages, each page was exactly one language. There are 6 subjects (Arts, Business, Humanities, Mathematics and Statistics, Science and Technology, Social Sciences), each page is some subset of these six.

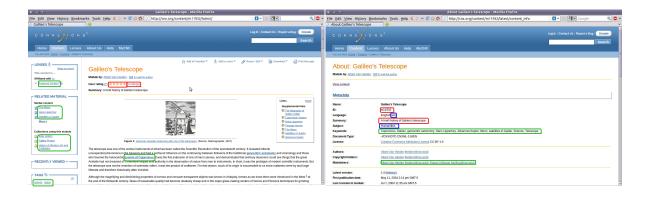


Figure 1: Example module: "Galileo's Telescope" and "About Galileo's Telescope". Red boxes are for integer or binary results, green boxes are for variables which have network structure, blue boxes are for factor variables which can also be used for network structure.

Because the network is so large, I cannot reasonably try to estimate the node ratings based on usual network models such as exponential random graph models. Instead what I propose is using the network structure to build a feature which assigns a single value to each node and then use that feature to help predict the ratings.

The feature that I will get from this network is similar to the PageRank algorithm which assigns a single value to each node via a random walk on a network.

## 2 Making Networks into Features

For this project I learned how to use Python (since that seemed like the easiest language to scrape data with), and I learned how to use the necessary libraries in Python (lxml and urllib2) to scrape the data. I got permission from cnx.org to harvest data from their website. I collected the data (making 18,219 files each 22 lines long – one file for each page, one line for page rating and for each of the 21 variables above). I imported all this data into R for statistical analysis.

I constructed the data matrix pageData with n=18,219 by p=70 matrix in which the columns are rating, number of ratings,  $\log(\text{total views})$ ,  $\log(\text{week views})$ , binary for art, binary for business, binary for math, binary for humanities, binary for sciences, binary for social sciences, number of lenses, number of collections the module is a part of (0 for pages that are themselves collections), number of key words, number of tags, number of related modules, number of links, number of cnx links, number of images, number of revisions, a column of 1s for an intercept, 50 columns of indicator functions for different languages. The number of total and weekly views is converted to a log scale to make those distributions approximately normal instead of very skewed to the right.

Then I made an adjacency matrix for each of the links and for the related modules, and bipartite adjacency matrices for the collections, tags, maintainers, key words, subject, and language. For the adjacency matrices, I used the PageRank algorithm (Page et al. 1998) on each of them to get new vectors of length n. For the bipartite adjacency matrices, I used the PageRank algorithm on AA' (because after two steps on the adjacency matrix, we are back on the original pages, so taking the PageRank algorithm on this gives the appropriate rankings). We cannot store AA' in memory, but to use the PageRank algorithm we do not need to if we write it as  $b = (1 - \beta)(AA')b + \beta \mathbf{1} = (1 - \beta)A(A'b) + \beta \mathbf{1}$  which is all vector times sparse matrix multiplication. In all of these cases I used the parameter in the PageRank algorithm of  $\beta = 0.15$ . I noticed that many of these PageRank values were discrete on some few number of values (see Figure 2) which makes sense since for example, related modules is a sparse graph with degree strictly less than 4 (and mostly isolated nodes). This by itself doesn't tell much about how users will surf these pages, but when it is combined with the other adjacency matrices then it becomes more relevant.

To alleviate this problem, I created one more adjacency matrix called Mixed where I blended together each of the other 8 matrices. Specifically, I made each of the previous graph matrices stochastic and then summed them together and approximated the first eigenvector of this resulting matrix (as in the spirit of PageRank). We don't need to worry about getting trapped anywhere on this resulting weighted graph since every page is in one language, and each language has at least one page in each subject, so we do not need to use the PageRank  $\beta$  value

# PageRank values for different Adjacency Matrices

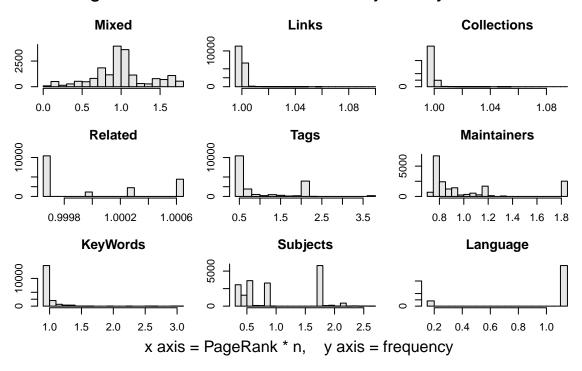


Figure 2: Histograms of the PageRank values according to each of the network structures and the first eigenvector of the Mixed network.

to make sure we have ergodicity. This resulting matrix is not sparse, and it cannot be stored in memory. This is not a problem though because of the associative property of vector multiplication on matrices, we are able to still calculate the first eigenvector by keeping each of the matrices separate. In other words we solve b = Ab by instead solving  $b = Ab = (A_1 + A_2 + \cdots + A_8)b = A_1b + A_2b + \cdots + A_8b$ . The right hand side of that equation we have each of the  $A_i$  is sparse so this is easy to do. So I constructed an n by 9 matrix of the PageRank weights for each of the pages based on these network structures. For ease of interpreting these numbers, for the rest of the paper I will normalize these vectors to each have a sum of n instead of a sum of 1.

# 3 Modeling with Network Features

The goal in this data set is to predict the rating of the pages. In the previous section, I explained the 70 features (rating, intercept, 12 positive integer features, 6 subject binary features, 50 language binary features) that are easily gotten from cnx.org plus the additional 9 features that I constructed from the network structure. In this section, I explore how those additional features improve our predictions in several different models. Although we are primarily interested in estimating the rating of each page we could also try to estimate the number of weekly or total views for each page since this is not missing any data whereas we only have 543 modules with any rating. More information about any of the following models can be found in Hastie et al. (2009).

In Python, I have about 270 lines of code to gather the data. In R I have about 240 lines of code to put the data into proper formats and to get the PageRank data. In R I have about 330 lines of code to analyze the data. Please email me at ahead@stanford.edu if you are interested in seeing the code and the output from R.

#### 3.1 Linear Regression

We do model selection here using forward-backward stepwise model selection based on the Akaike information criterion (AIC). Features with a \* indicate a p-value less than  $10^{-3}$ . Features with a \*\* indicate a p-value less than  $5 * 10^{-5}$ 

When we do linear regression without the network features we include the following variables: \*\*week views,

## Page Ratings vs Predicted Page Ratings LM without networks

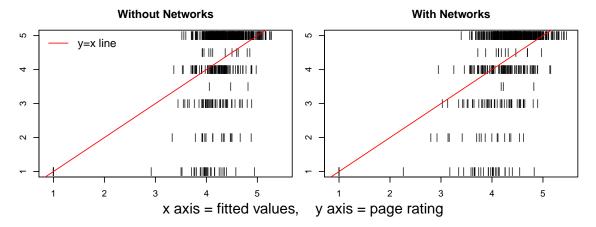


Figure 3: Fitted values excluding and including network structure variables.

business subject, subject social sciences, number of collections, number related modules (negative), number of images (negative), English, French (negative), Macedonian (negative), Portugeues, Chinese (negative). We get an adjusted  $R^2$  value of 0.128 and an AIC of 2377.

When we do linear regression with the network features we include the following variables total views (negative), \*week views (negative), business subject, humanities subject, \*subject social sciences, number of images (negative), French (negative), Macedonian (negative), Portugeues, Chinese (negative), \*\*Mixed network, \*Collections network, \*Tags network (negative), \*Maintainers network, Key Words network. We get an adjusted  $R^2$  value of 0.1801 and an AIC of 2347.

In both cases, the residuals are skewed to the left, but the set including network data is noticeably less skewed to the left (see ratings vs fits in Figure 3). and we do a much worse job of predicting the low ratings nearly correctly than we do at predicting high ratings (recall, we have far fewer low ratings). In general, it seems as though the network structures are useful in helping to predict Rating.

From the perspective of the authors of those pages, it might be more interesting to try to predict page views from these features. If instead of trying to predict rating we try to predict the total views (and we do not include weekly views), then when we regress on the other features without networks, our adjusted  $R^2$  value is 0.3497 and AIC is 1963 whereas with networks it is 0.4179 and AIC is 1948.

#### 3.2 Generalized Linear Models

If we think of low ranks as rare events (which seems reasonable given the percentages of each rating) then we can model 5 - the rating as a poisson distribution and do a generalized linear model using the poisson exponential family. Then we can again do a forward-backward stepwise selection to select which variables to include based on AIC. The resulting model without networks gives an AIC of 2114 and with networks gives an AIC of 2050.

Since almost two thirds of these are rated as 5 stars, we might use a binomial family to see if we can correctly predict which modules are 5 stars. When we do this using forward-backward stepwise selection based on AIC, then we need to exclude languages from consideration since some languages did not contain both scores of 5 and scores of less than 5. Without networks (or languages) we get an AIC of 672. With networks (without languages) we get an AIC of 646. Note that these AIC values are much lower since we are only considering the binary response of 5 stars or less than 5 stars.

## 3.3 Sparse Linear Regression via LASSO

Using the LASSO method to do sparse linear regression to predict ratings. We find the value where features where Mallows Cp makes a sudden drop or starts to increase. Because there are some language features that only have only a single value for all the ratings, we cannot include language in these models. Also, in the model including networks, we should remove the number each of the network items that we had been including before because otherwise, we get a very large positive coefficient for the number of related pages and the PageRank network values for Related Modules (in Figure 2 we see that there are only a few values that the PageRank network values for

Related Modules can take on). This suggests that the number of related modules is very closely related to the PageRank values we get for Related Modules and we should not be including both of those features in the model. After removing these features, we see that for the model without network structures, the best choice (based on Mallows Cp) is clearly to choose 10 non-zero coefficients. For the model with network structure, it looks as though the model with 10 features is a good candidate to consider. The model without network structure that includes exactly 10 features has a residual sum of squares of 2556 and an  $R^2$  of 0.9516. The model with network structure that includes exactly 10 features has a residual sum of squares of 2496 and an  $R^2$  of 0.1166.

#### 3.4 Recursive Partitioning and Regression Trees (rpart)

This method does not work very well for this particular data set for either the models without networks or the model with networks. Using an ordinal response splitting method for the rpart function in R is more appropriate than just using a categories for the rating. When we use the ordinal rpart we have an error rate of 1 - 354/354 = 0.348 without using network structure and 360/543 = 0.337 using network structure. Neither of these are of any practical significance though considering if we always guess 5 stars then we have an error rate of 341/543 = 0.373.

#### 3.5 Support Vector Machines

This method also does not work very well for this particular data set for either the models without networks or the model with networks. In both the models without networks and with networks, SVM always predicts a 5 with a radial. With a linear kernel we run a 5 fold cross validation and get a test error rate of 0.370 for the model without networks and 0.368 for the model with networks.

We can change this to a binary prediction of 5 or less than 5. In this case, using a radial kernel we minimize the error rate with respect to the tuning parameter gamma. When we do this, we get a test error rate of 0.348 for the model without networks and 0.327 for the model with networks. With a linear kernel we run a 5 fold cross validation and get an error rate of 0.331 for the model without networks and an error rate of 0.317 for the model with networks.

#### 4 Discussion

The purpose of this paper is to demonstrate that measures of network structure which assign values to the nodes can be used in machine learning problems. Although I opted to use the PageRank algorithm on the cnx.org networks there may be many other interesting values to choose to assign either real numbers (such as betweenness, centrality, or influence) or factors (such as partitions, connected components, or community structure).

In all of these examples we do the same model selection process on both the features that include the network structure and the features that do not. In every model, we either have that the model is either inappropriate for the data or we have that the features including network structure do better than the features excluding network structure.

With a more in depth exploration of which variables we are using in each model we can see that including the Mixed network adds information to the model that the features from individual adjacency matrices cannot. I constructed the Mixed network in a somewhat arbitrary combination of the different adjacency matrices though. In future work, it would be interesting to examine how we could add these networks together with different weights, and possibly try to find a way to optimize these weights.

#### References

Hastie, T., Tibshirani, R. & Friedman, J. (2009), The elements of statistical learning: data mining, inference and prediction, 2 edn, Springer.

**URL:** http://www-stat.stanford.edu/ tibs/ElemStatLearn/

Page, L., Brin, S., Motwani, R. & Winograd, T. (1998), 'The pagerank citation ranking: Bringing order to the web'.