Predicting the Internet's Evolution with Decision Trees and Lasso Logistic Regression Models

Cuong Nguyen '14, Grinnell College

Abstract

The Internet self-evolves rapidly and its dynamic structure poses many interesting questions for researchers in network analysis. In this paper I show how we can simplify the entire Internet as a mathematical graph and then extract its structural characteristics; these characteristics in turn help us build statistical models that can predict how the Internet will evolve. The data describing the Internet structure are both clustered and unbalanced. I hence test various models, including lasso logistic regression, gradient-boosted decision trees and random forest decision trees, to see how well they cope with unbalanced and clustered data. The best performing model was created through a gradient-boosted decision tree that balances flexibility in fitting with robustness in prediction. I show that we can achieve good predicting power using fairly simple explanatory variables, but I also discuss how we can extract more sophisticated variables to improve the models' performance.

1. Background and significance:

Networks are in every corner of our lives. Our brains are a network of neurons and our biological traits are the results of interactions in our genetic network. We participate actively in social networks and advance knowledge by contributing to research/patent networks.

As a result, researchers have placed a strong interest in understanding networks. In this paper, I build statistical models to study one of the most important networks: the Internet. Although humans created the Internet, it has since evolved so freely that we are still exploring its dynamic structure. Collecting data to map the Internet always poses a challenge and it is only recently that we can study empirical models of the Internet.[1] Previously, researchers were interested in studying the Internet as a static network. Currently, the focus has moved to modeling how the Internet evolves.[2] My work furthers this effort by developing and evaluating models to predict the Internet's evolution over time.

2. Methods

a. Data collection & graphical representation of network

Data for this project were collected by The Cooperative Association for Internet Data Analysis (CAIDA). A node is a local network of computers under control of one administrator (called an Autonomous System, AS). For instance, CAIDA sees Dordt College as one node.

Each edge connects two nodes, and its direction (FromNodeID, ToNodeID) shows how information flows. Each edge has a weight of either 0 or 1, meaning the connection is either free (peer-to-peer) or has a cost (provider-customer).[3] The first row in Table 1, *2-9-1,* represents a connection going from node 2 (N_2) to node 9 (N_9) and its weight is 1.

This data set contains information on 49,083 edges and 21,861 nodes for 42 distinct snapshots. Each snapshot represents one time period during 2004-2006. Each snapshot at period t consists of three columns of data that can be modeled as a mathematical graph G_t, t = 1,2,...42.

My hypothesis is that *prior behavior of the Internet's structure signals us how it will likely behave in the future.* Specifically, let G_T denote the graph of Internet at time t = T; we can use historical characteristics of edges in G_1, G_2...G_{T-1} to predict whether those edges remain in G_T. In other words, we can predict how the Internet evolves into G_T by using the edges' historical characteristics as explanatory variables.

Table 1: Raw data

FromNodeID	ToNodeID	Weight
2	9	1
9	1	0
…	…	…

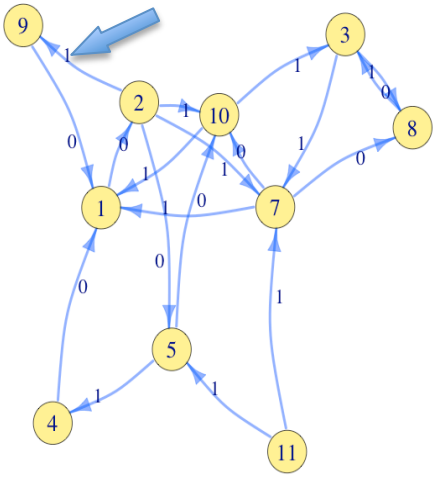


Figure 1: Graphical representation of the Internet in one snapshot of time.

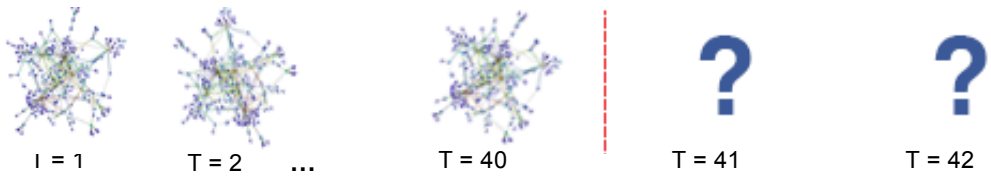


Figure 2: The Internet's evolution as a series of graphs.

b. Variable creation

Applying graph theory, I extracted multiple characteristics of the Internet for a specific time period. For example, if our goal is to predict the structure of the network at t = 41, the dependent variable Y is binary and indicates whether a given edge that existed in G_{40} survives into G_{41}. Our purpose is to predict Y from all explanatory variables X_i.

When predicting the structure of G_{41}, we used 56 explanatory variables X_i, i=1…56. Variables (X_i, i=1…39) indicate whether an edge exists in the first 39 graphs . For instance, if the edge *2-9* exists in G_{39} but not in G_{38}, then X_{39}=1 and X_{38}=0.[4] The variable X_{40} is the weight of each edge that exists at t = 40.[5] This process results in 40 explanatory variables (columns) where each row represents a particular edge that existed in G_{40}.[6]

More complex quantitative explanatory variables were also created. Variables X_{41} to X_{44} indicate the in- and out-degrees of the two nodes associated with each edge. An edge always has a *From-Node* and a *To-Node*; for example, edge *5-4* in figure 1 has node N_5 as *From-Node* and node N_4 as *To-Node*. For the *From-Node* N_5, two edges come in and two edges come out, so it has an in-degree X_{41} = 2 and an out-degree X_{42} = 2. For the *To-Node* N_4, one edge comes in and one edge comes out, so it has an in-degree X_{43} = 1 and an out-degree X_{44} = 1. Variables X_{41} to X_{44} are calculated based on data in G_{40}. Variables X_{45}-X_{48} are calculated similarly to X_{41}-X_{44}, but use lagged data in G_{39} instead of G_{40}.

Variable X_{49} is the percentage of the number of '1's in series X_1…X_{39} of each observation. Variable X_{50} counts how many times the binary series X_1 to X_{39} switches between 0 and 1. Variable X_{51} looks backward from X_{39} to X_1 until it reaches the last "1" in the series.[7] Variables X_{52} and X_{53} represent the Closeness of *From-Node* and *To-Node* of an edge. Closeness measures how close a node is relative to the center of the graph. Variable X_{54} indicates each edge's Betweeness in the graph by counting how many shortest paths cross it. Variables X_{55}-X_{56} are the "ranks" of *From-Node* and *To-Node* in the network.[8]

In summary, I used the given 40 three-column tables (similar to Table 1), representing the internet at 40 time periods to create a dataset consisting of 56 explanatory variables. This new dataset was then used to predict the existence of an edge at G_{41}.

c. *Methodology: cluster analysis and model selection*

While I was analyzing the data, scatter plots showed multiple clusters in the data. I also noticed that edges have a strong tendency to continue to exist. In other words, P(Y=1) was always much higher than P(Y=0). In fact, 94% of the edges in G_{40} survive into G_{41}. Samples of these plots are included in Appendix 1 and stress the two main problems of my data: 1) the X_i space has many clusters and 2) the response Y is very unbalanced.

To predict the existence of an edge in G_{41}, I built predictive models in three main classes:

- LASSO logistic regression[9]: LASSO logistic regression is a variable selection method that is used to select a parsimonious set of explanatory variables for the efficient prediction of a response variable. It minimizes the residual sum of squares subject to the condition that the sum of the absolute value of the coefficients is less than a constant.

- Gradient Boosting Machine (GBM) decision trees: GBM is a series of trees built consecutively to capture the signal in X_i slowly but robustly.

- Random Forest (RF) decision trees: RF is an aggregation of many parallel, de-correlated trees (hence the name "forest") that favors stronger predictors.[10]

3. Results

Table 2 demonstrates that the survivor nodes (Y=1) have a higher weight, percentage of '1', to-indegree and a lower count of switches, from-indegree, to-indegree and to-outdegree. In logistic regression, the pseudo McFadden R^2 is 13%. The model is statistically significant at 1%. Among the explanatory variables, all in-, out-degrees (X_{41} to X_{44}) and their lagged (X_{41} to X_{44}) are significant. Weight (X_{40}), percent of '1' (X_{49}), count of switches (X_{50}), time until last 1 (X_{51}) and ranks (X_{55} & X_{56}) are also significant. Most of the predictors from X_1 to X_{37} are not significant. In tree models, an analysis of relative importance also confirms that in-, out-degrees, percent of '1', count of switches, time until last 1 and ranks are strong predictor. Appendices 3 and 4 show selective outputs of the logistic and tree models.

Table 2: Summary Statistics of Several Explanatory Variables

Y	Stats	Weight (X_{40})	Percent of "1" (X_{49})	Count of Switches (X_{50})	from-In-degree (X_{41})	from-out-degree (X_{42})	to-In-degree (X_{43})	to-out-degree (X_{44})
Y=0	Mean	0.7	0.5	3.2	61.6	36.3	289.3	47.6
(2087 obs.)	S.d.	0.5	0.3	3.0	145.7	81.1	535.9	85.0
Y=1	Mean	0.8	0.7	2.1	37.3	19.0	487.0	37.9
(46996 obs.)	S.d.	0.4	0.3	2.3	141.2	63.7	696.1	71.5

After creating prediction models, I divided my data into training data and testing data. I fit the models using training data and assess the performance of these fitted models using testing data.[11] I use the *Area Under the receiver operating characteristic Curve* (AUC) as my main criteria to select models. A detail explanation of AUC is included in Appendix 2. AUC is similar to R^2 for a linear model as it can tell us how well the models capture the variation in the data. The main advantage of AUC is that it allows us to compare linear to non-linear models, a feature R^2 lacks.

Table 3 represents the AUC scores of my models. For the training scores, the logistic model performs most poorly, suggesting that its linearity is rather rigid and cannot explain the variation in the data very well. Random Forest tree performs the best with an AUC of 0.98, indicating that this model captures the variation quite well. However, an almost-perfect AUC score (near 1) may also signal that it might over-fit to the noise of the training data.

Table 3: Models performance

AUC	Logistic	GBM tree	RF tree
Training	0.75	0.83	0.98
Testing	0.73	0.76	0.73

Testing data is used to re-assess the performance of the models. After I fit my models on training data, the testing scores indicate how well these fitted models perform on a different dataset. Logistic model's AUC drops from 0.75 to 0.73, predicting quite consistently with new data. In contrast, the AUC of RF tree drops dramatically from 0.98 to 0.73 – a strong indicator that it over-fits to the noise in training data. Among the three models, GBM tree offers the best balance: it fits training data well (training AUC=0.83) but also predicts robustly on new testing data (testing AUC=0.76).

4. Discussion

In this paper I compare three methods of predicting the survivorship of edges in the graphs of the Internet. Each of these models can be used to predict the future structure of the Internet. In Table 3, we observe an interesting trade-off between flexibility (training AUC) and robustness (testing AUC). Moreover, the Gradient-Boosted tree model has the best balance of flexibility and robustness when dealing with unbalanced and clustered data.

However, my findings are subject to several limitations. My best testing AUC is 0.76, which is an encouraging result but still needs improvement. Any model with AUC > 0.5 has prediction power, but future work could include more snapshots and more explanatory variables to improve the predictive power of my models. I created only 56 simple predictors, but my data has N = 49,083 samples. Clearly there are enough degrees of freedom to extract and use more sophisticated predictors from the historical graphs. Extracting more sophisticated predictors, however, requires a deeper understanding of graph theory and computer network theory. While my results are limited, there is tremendous opportunity for further research in this area, research that could dramatically increase our ability to make predictions in complex networks.

---

[1] Gao (2001), Xia & Gao (2004) and Lyon (2003).

[2] Leskovec et al. (2005 & 2007).

[3] Gao (2001), Xia & Gao (2004) explains how they collect the AS dataset using the Border Gateway Protocol method.

[4] Edge names are unique in the AS-CAIDA data set, so a simple search suffices to create X_1 to X_{39}.

[5] I use the R package *igraph*, Csardi & Nepusz (2006) to extract these explanatory variables.

[6] In creating our predictive models, we made the assumption that the previous 40 time periods were sufficient to model the next time periods. To predict G_{41} we used G_1 through G_{40}. If we were to predict G_{101} we would use G_{61} through G_{100}.

[7] For example, a simple series: 0, 0, 0,…, 1, 1 has $X_{51} = 2$.

[8] Closeness, betweeness and page rank are explained in Freeman (1979) and Csardi & Nepusz (2006).

[9] See Tibshirani (1996) for a full description. When I apply the LASSO technique, it suggests neglecting variables from X_1-X_{37}. A possible explanation is that percent of '1' (X_{49}), count of switches (X_{50}) and time until last 1 (X_{51}) have already capture most of the variations in X_1-X_{37}.

[10] Details of GBM and RF are explained in Hastie et al. (2013), Ch. 10 and implemented in Ridgeway (2007).

[11] Hastie et al. (2013), Ch. 7 reviews model assessment using separate training and testing data.

References

Aggarwal, V., Feldmann, A., & Scheideler, C. (2007). Can ISPs and P2P users cooperate for improved performance?. *ACM SIGCOMM Computer Communication Review*, *37*(3), 29-40.

AS Relationships. (n.d.). *AS Relationships*. Retrieved May 6, 2014, from http://www.caida.org/data/as-relationships/

Csardi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal, Complex Systems*, *1695*(5).

Freeman, L. C. (1979). Centrality in social networks conceptual clarification.*Social networks*, *1*(3), 215-239.

Gao, L. (2001). On inferring autonomous system relationships in the Internet.*IEEE/ACM Transactions on Networking (ToN)*, *9*(6), 733-745.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., & Tibshirani, R. (2009). *The elements of statistical learning* (Vol. 2, No. 1). New York: Springer.

Leskovec, J., Kleinberg, J., & Faloutsos, C. (2005, August). Graphs over time: densification laws, shrinking diameters and possible explanations. In*Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (pp. 177-187). ACM.

Leskovec, J., Kleinberg, J., & Faloutsos, C. (2007). Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *1*(1), 2.

Lyon, B. (2003). The opte project. *The Opte Project*. http://opte.org/maps/tests/

Ridgeway, G. (2007). Generalized Boosted Models: A guide to the gbm package. *Update*, *1*(1).

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267-288.

Xia, J., & Gao, L. (2004). On the evaluation of AS relationship inferences [Internet reachability/traffic flow applications]. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE* (Vol. 3, pp. 1373-1377). IEEE.

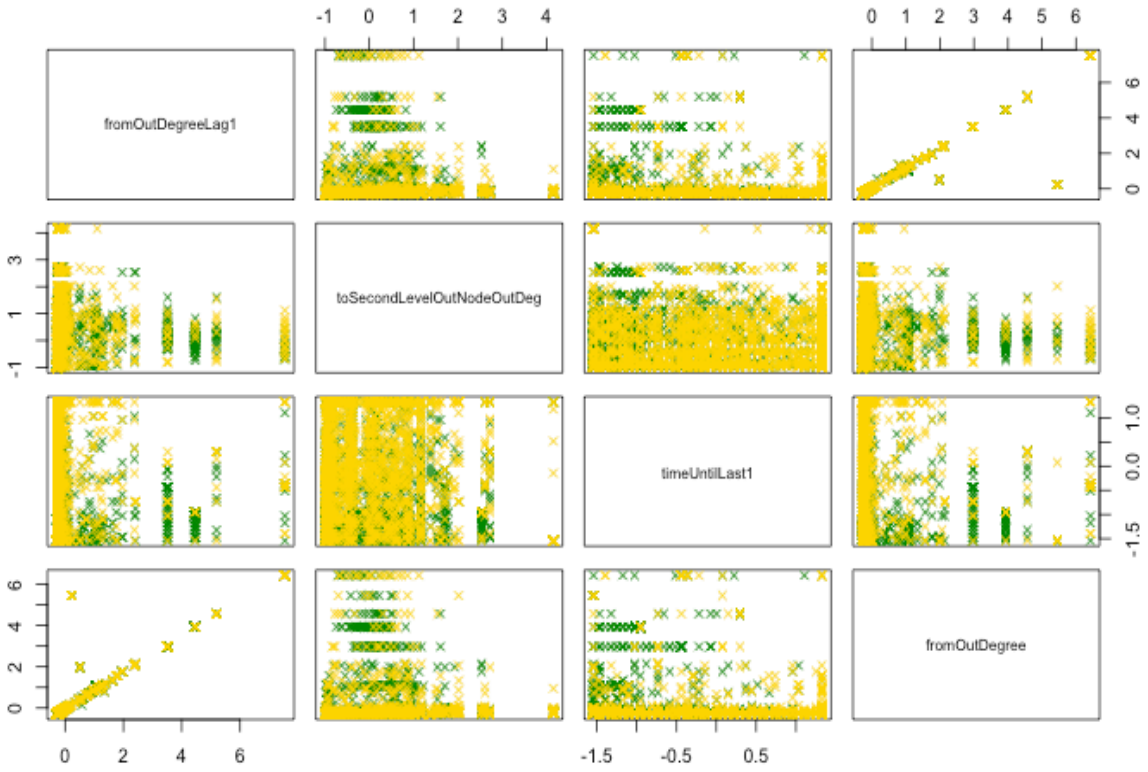
Kaggle's Facebook competition: https://www.kaggle.com/c/facebook-ii

Appendix 1: Selective pair plots of explanatory variables

Response:

Green: Y=0

Yellow: Y=1



Appendix 2: Area Under the receiving operator Curve (AUC)

AUC is a commonly used evaluation method for binary choice problems, which involve classifying an instance as either positive or negative. Its main advantages over other evaluation methods, such as the simpler misclassification error, are:

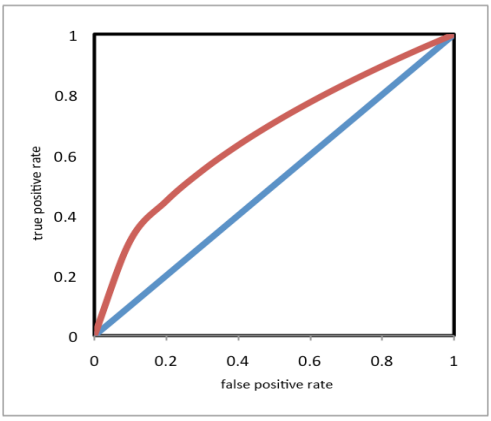
1. It is insensitive to unbalanced datasets.

2. For other evaluation methods, a user has to choose a cut-off point above which the target variable is part of the positive class (e.g. a logistic regression model returns any real number between 0 and 1 - the modeler might decide that predictions greater than 0.5 mean a positive class prediction while a prediction of less than 0.5 mean a negative class prediction). AUC evaluates entries at all cut-off points, giving better insight into how well the classifier is able to separate the two classes.

Understanding AUC

To understand the calculation of AUC, a few basic concepts must be introduced. For a binary choice prediction, there are four possible outcomes:

- True positive - a positive instance that is correctly classified as positive;
- False positive - a negative instance that is incorrectly classified as positive;
- True negative - a negative instance that is correctly classified as negative;
- False negative - a positive instance that is incorrectly classified as negative);



The true positive rate (*sensitivity* or *recall*), is calculated as the number of true positives divided by the total number of positives. When identifying aircraft from radar signals, it is proportion that is correctly identified.

The false positive rate (equivalent to *1 - specificity*) is calculated as the number of false positives divided by the total number of negatives. When identifying aircraft from radar signals, it is the rate of false alarms. If somebody makes random guesses, the ROC curve will be a diagonal line from (0,0) to (1,1) - see the blue line in the figure below.

For example: somebody who randomly guesses that 10 per cent of all radar signals point to planes. The false positive rate and the false alarm rate will be 10 per cent. Somebody who randomly guesses that 90 per cent of all radar signals point to planes. The false positive rate and the false alarm rate will be 90 per cent. Meanwhile a perfect model will achieve a true positive rate of 1 and a false positive rate of 0.

While ROC is a two-dimensional representation of a model's performance, the AUC distils this information into a single scalar. As the name implies, it is calculated as the area under the ROC curve. A perfect model will score an AUC of 1, while random guessing will score an AUC of around of 0.5. In practice, almost all models will fit somewhere in between.

Appendix 3: Selective regression outputs from logistic regression

Call: glm(formula = Y ~ . - X1, family = binomial, data = train)

Coefficients:

	Estimate	Std. Error	z value	Pr(>|z|)	Significance
(Intercept)	0.824	0.116	7.080	0.000	***
X38	0.390	0.159	2.443	0.015	*
X39	0.333	0.159	2.095	0.036	*
weight	0.716	0.079	9.073	< 2e-16	***
fromInDegree	0.022	0.006	3.413	0.001	***
fromOutDegree	-0.031	0.009	-3.395	0.001	***
toInDegree	0.010	0.003	3.577	0.000	***
toOutDegree	-0.012	0.004	-3.078	0.002	**
fromCloseness	0.029	0.014	2.062	0.039	*
toCloseness	-0.004	0.024	-0.172	0.864	
betweeness	-0.011	0.020	-0.540	0.590	
percentOf.1.	3.821	5.240	0.729	0.466	
countOfSwitch	-0.092	0.011	-8.631	< 2e-16	***
timeUntilLast1	0.018	0.004	4.790	0.000	***
fromInDegreeLag1	-0.023	0.006	-3.610	0.000	***
fromOutDegreeLag1	0.026	0.009	3.004	0.003	**
toInDegreeLag1	-0.010	0.003	-3.757	0.000	***
toOutDegreeLag1	0.009	0.004	2.519	0.012	*
fromrank	0.472	9.790	4.818	0.000	***
torank	63.930	12.800	4.995	0.000	***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 17265 on 49082 degrees of freedom

Residual deviance: 15060 on 49027 degrees of freedom

AIC: 15172

Appendix 4: Selective outputs from Gradient Boosted tree
> summary (GBM.model.final)

varriables	relative importance
percentOf.1.	13.30
fromInDegreeLag1	8.68
betweeness	8.30
countOfSwitch	6.16
timeUntilLast1	5.88
torank	5.42
toInDegree	5.28
fromInDegree	4.68
toOutDegree	4.39
toOutDegreeLag1	4.33
fromrank	4.09
toInDegreeLag1	3.52
fromOutDegreeLag1	2.93
fromCloseness	2.27
toCloseness	1.84
X38	1.76
fromOutDegree	1.72