Link Prediction for Facebook Wall Post

Chris Li, Jiaqi Li, Jaya Ren
University of California, Los Angeles

Jun 10th, 2022

1 Abstract

Link prediction problem has been a heated topic in the field of network analysis. Given timestamp information and a network of interactions between different users, we are interested in learning whether it is possible to predict if a new interaction will exist between two users in the near future. We examined different network-related metrics and compared the performance of multiple machine learning models on our data. We showed that under certain assumptions about the data, several machine learning models are able to achieve a fair level of accuracy in predicting new interactions.

2 Introduction

Over the decades, researchers have been working on problems related to online social networks and trying to get a deeper understanding of this relatively new form of community. One important feature of social interactions is that the communications are temporal, i.e., each interaction (link) is associated with a specific time. Jiali Lin et al. explored the differences and similarities of user interaction patterns on two social media platforms in China by characterizing temporal interaction behaviors using the HMM (Hidden Markov Model) and clustering algorithms (such as self-organizing maps) [6]. Another paper studied in detail the structure of multiple online social media networks and confirmed that properties such as power-law and small-world are generally shared across online platforms [7].

Beside learning about the nature of online social networks and user interactions, researchers are interested in whether future connections can be predicted based on prior information about the network. Some researchers evaluated different metrics as the link predictors which based solely on network topology which predict whether a node will gain or lose connection with another node [3]. Others made use of machine learning models with network metrics as inputs. For example, a paper in 2010 developed an unsupervised machine learning model to estimate relationship strength from interaction activity and user similarity [10]. Recent studies have incorporated more complex structure like tie-decay to allow network analysis in continuous time rather than discrete time [1].

After closely reading the above scholarly papers regarding the overall structure of online social networks and current methods for predicting future interactions, we also learned from several other sources that provided important methodology for approaching our problem specifically. A study on the evolution of user interaction in Facebook provides an idea for analyzing temporal networks, that is, by considering several snapshots (interaction data within a certain interval of time) [9]. We can learn from this paper to use the resemblance metric for comparing and evaluating the dynamics across different snapshots.

From the paper written by researchers David Liben-Nowell and Jon Kleinberg, we learned how to evaluate different metrics as the link predictors which based solely on network topology [5]. Similarly, Michael Fire et al. explained their method of building accurate classifier for identifying missing links, from which we learned about structural features of social networks [2]. Since our project involves machine learning, used methods for improving the "accuracy" of machine learning models such as feature selection and ensemble methods [8]. Lastly, we also learned about the Node2Vec model, a model specifically designed for network analysis that transform nodes into vectors [4]. However, the Node2Vec algorithm did not outperform the other commonly seen network-related metrics.

3 Materials and Methods

3.1 Exploratory Data Analysis

The data set we use is the Facebook Wall Post Network (facebook-wosn-wall). This dataset is publicly available and contains data collected from about 2005 to 2009. There are in total 274,086 edges and 46952 nodes in this data. Each node represents a user on Facebook, and a directed edge from node i to node j means that user i made a post on user j's personal Facebook page. There is also a timestamp information associated with each edge, which we will make use of for our temporal analysis.

We decided to partition the data into two graphs according to time and take the first 80% of the edges as our training data. It is possible to partition the graph into more than two parts, but we chose not to do that for simplicity purpose. Specifically, we noticed that there was a huge increase in user interaction from July, 2008 to January, 2009 (potentially due to Facebook updating its app and introducing new features).

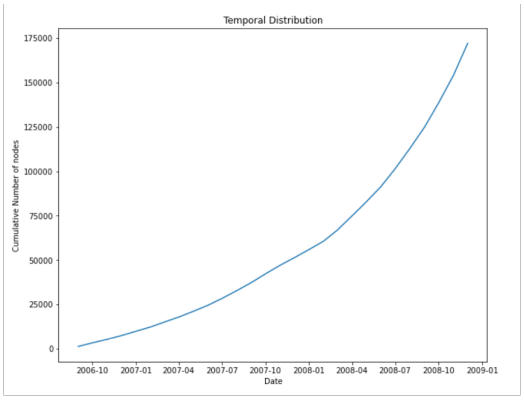
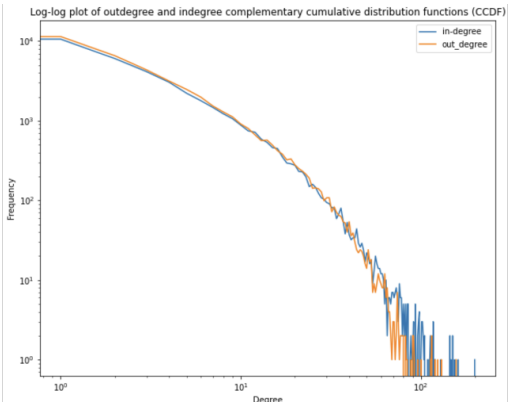
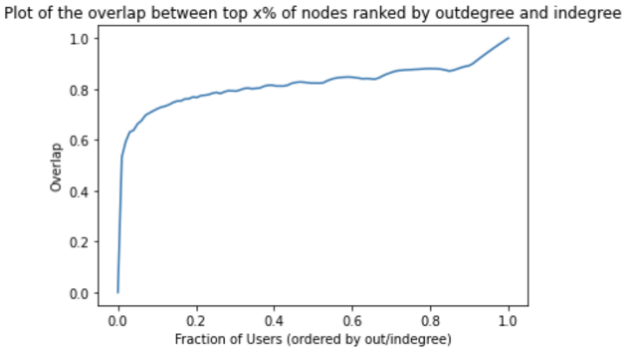


Figure 1: Temporal Distribution of Edges in the Facebook Wall Post Data

After separating our data, we explored the training data and noticed that the graph has a giant component that contains more than 99% of its total nodes, meaning that most users are connected in some sense to the larger community. We further explored that more than 95% of the users have in-degree and out-degree less than 25, while only very few users have extremely high degree (ie. above 100). We then calculated the percentage of overlap between nodes with highest in-degree and out-degree. Not so surprisingly, we found a large overlap, indicating that users who are actively making posts on other people's Facebook pages also receive relatively large numbers of comments on their own personal pages. Based on this observation, we decided to ignore the directed property of the network in this project and treat all the edges as undirected, because we believe it is reasonable that the interaction can be treated as two-way.



(a) In-degree and Out-degree of Nodes



(b) Percentage of Nodes Overlap

3.2 Feature extraction

Here we computed some features for machine learning models later to include more information about our graph.

- Preferential Attachment: Measure that is based on the concept that high degree nodes are more likely to connect than low degree nodes. It is defined as

$$|\Gamma(x)| \cdot |\Gamma(y)| \tag{1}$$

  where $\Gamma(x)$ is a set of neighbors of node x.

- Clustering Coefficient: Measure of the degree to which nodes in a graph tend to cluster together. For unweighted graphs, the clustering of a node u is the fraction of possible triangle through that node that exist, and it is defined as

$$c_u = \frac{2T(u)}{deg(u)(deg(u)-1)}, \tag{2}$$

  where $T(u)$ is the number of triangles through node u and $deg(u)$ is the degree of u.

- Common Neighbors: The number of common nodes that connect to both two nodes, and it is simply defined as

$$|\Gamma(x) \cap \Gamma(y)| \tag{3}$$

- Jaccard Index: Normalized number of common neighbors, and it is defined as

$$\frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{4}$$

- Adamic-Adar index: Measure that based on intuition that having a common friend with low degree is more significant that having one with high degree. This feature has been shown to be very informative for social networks. It is defined as

$$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|} \tag{5}$$

- Community Resource Allocation Index: Community detection measure defined as

$$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{f(z)}{\Gamma(z)} \tag{6}$$

  where f(z) is 1 if z belongs to the same community as x and y and 0 otherwise. This feature combines community detection and common neighbors features.

3.3 Data Preparation

Since we are making a binary prediction, we need to add negative examples to our training and testing data. We randomly chose 3 times of fake edges which means no edges between two nodes to add to our data, so we have nearly a million pairs of nodes in our data. Next, we added the features that we mentioned above to our data. After computing and adding features, we used the variance inflation factor to detect multicollinearity. Multicollinearity among independent variables will result in less reliable statistical inferences. From Table (a), we could see that 'Common Neighbors' feature is highly correlated to 'Adamic-Adar Index'. However, common neighbors didn't offer much extra information that Adamic-Adar Index already has. Therefore, we drop the common neighbors feature for data simplicity and avoiding multicollinearity in our data. Then, we transformed our data using normalization and principal component analysis (PCA). Normalization gives equal weigh to every variable, and PCA allows us to capturing the most important features.

	Features	VIF
1	Deg_Source	2.36
2	Deg Target	2.80
3	Clustering Coeff Source	1.20
4	Clustering Coeff Target	1.23
5	Common Neighbors	1623.36
6	Jaccard Coefficient	4.44
7	Adamic Adar Index	2985.02
8	Preferential Attachment	4.24
9	Resource Allocation Index	276.34

(a) VIF with Common Neighbor Feature

$\Longrightarrow$

	Features	VIF
1	Deg_Source	2.32
2	Deg Target	2.80
3	Clustering Coeff Source	1.20
4	Clustering Coeff Target	1.23
5	Jaccard Coefficient	4.44
6	Adamic Adar Index	9.97
7	Preferential Attachment	4.06
8	Resource Allocation Index	7.32

(b) VIF without Common Neighbor Feature

Table 1: Variance Inflation Factor

3.4 Model Fitting

After we transformed our numerical data through PCA, we added the two categorical predictors, whether a pair of nodes is in the same Louvain community and whether a pair of nodes is in the same greedy modularity community, back to our data. Then, we start fitting our training data with various machine learning models. As shown in Figure. , we employed many machine learning models, including logistic regression, SVM (support vector machine), GBM (gradient boosting machine), random forest, decision tree, LDA (linear discriminant analysis), QDA (quadratic discriminant analysis), and naive Bayes. We choose AUC (Area under ROC curve) as our criterion for model performance, which represents how well a model can distinguish between classes. For example, AUC = 0.7 means there is a 70% chance that the model is capable to distinguish between two labels.

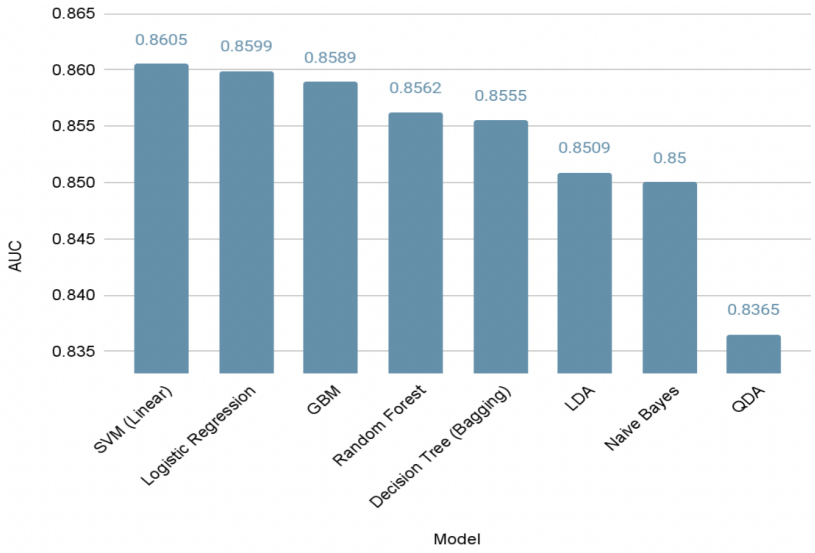
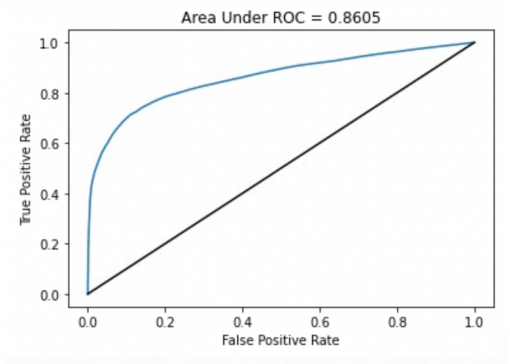


Figure 3: Area Under ROC Curve for Testing Data Comparison

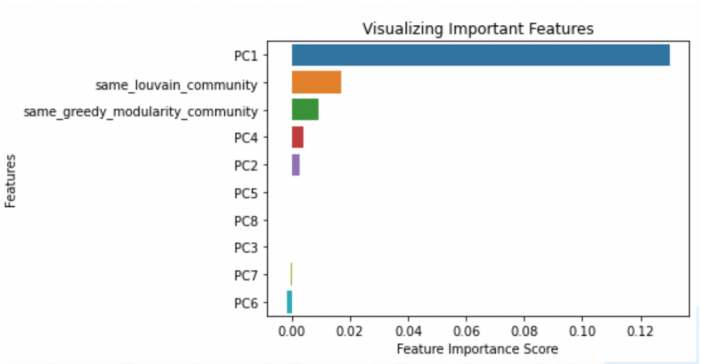
Shown in Figure. 3, all of our models except for QDA achieved over 85% AUC for testing data. Our best three models are SVM with linear kernel, logistic regression, and GBM that all achieved about 86% AUC. Now we take a closer look at these three models.

3.4.1 SVM with Linear Kernel

SVM with linear kernel is our best model. Shown in Figure. 4a, SVM's AUC for testing data is 0.8605. The most important three features are PC1, same louvain community, and same greedy modularity community, according to Figure. 4b. Recall that from PCA, PC1 itself explained over 46% of the variation in the numerical part of the training data, and the first four principal components combined explained over 87% of the numerical part of the training data, which is why we see the importance of

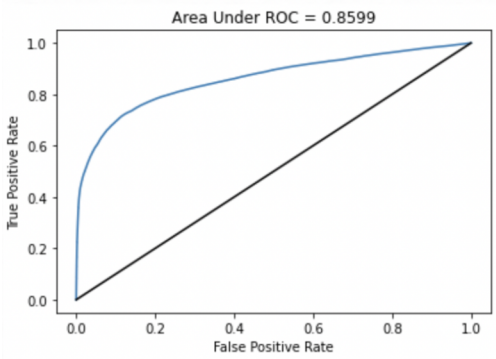


(a) SVM AUC

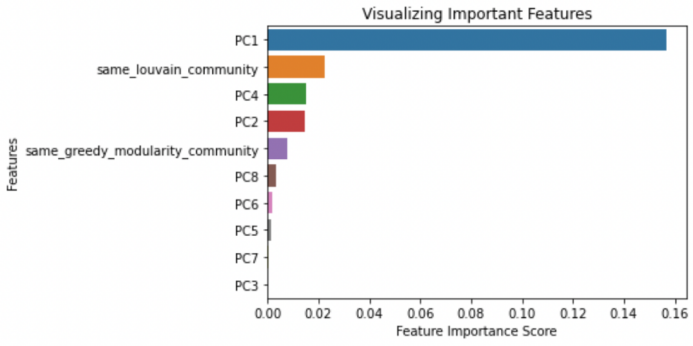


(b) SVM Feature Importance

Figure 4: SVM Model with Linear Kernel



(a) Logistic Regression AUC



(b) Logistic Regression Feature Importance

Figure 5: Logistic Regression Model with Linear Kernel

most of other principal components got squashed to zero.

3.4.2 Logistic Regression

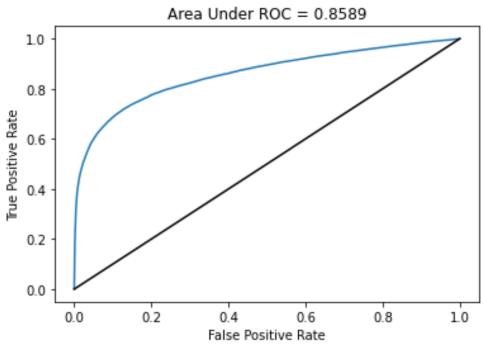
Our second best model is logistic regression. This is a full model without tuning any parameter, and its AUC is 0.8599 shown in Figure. 5a. Based on Figure. 5b, PC1 and same louvain community are still the most important predictors, followed by PC4, PC2, and then same greedy modularity community.

3.4.3 GBM

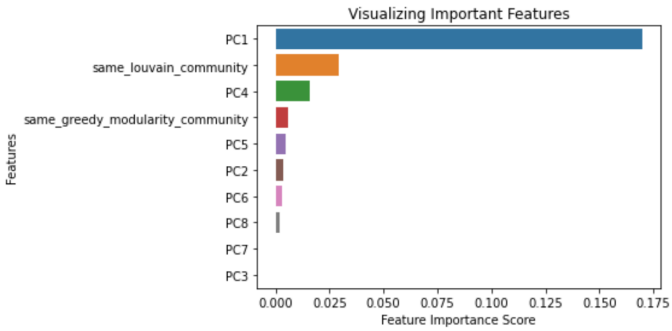
The third best model, GBM, achieved an AUC of 0.8589 shown in Figure. 6a. We observe consistent pattern that PC1 and same louvain community are the most important two features, followed by PC4 and same louvain community, from Figure. 6b.

3.5 Model Improvement

We also used some common techniques to improve machine learning model score. First, we used GridSearchCV to find the most optimal parameter for logistic regression. We had an AUC of 0.8599 before applying GridSearchCV parameters, and an AUC of 0.8593 after applying GridSearchCV parameters. Since GridSearchCV could be very time-consuming for a large dataset, we only applied this on logistic regression. From the result, we could see that it didn't have much improvement, but a slightly declined on the score. We believed that the reason why the score didn't improve was because GridSearchCV found the best parameters based on the training data, not testing. So, the parameters might just not be the best parameters for the testing data. Second, we used boosting and bagging technique on the decision tree model, where boosting adjusts the weight of an observation based on last classification and bagging is a way to decrease the variance in the prediction. We had an original AUC of 0.8495, and an



(a) GBM AUC



(b) GBM Feature Importance

Figure 6: GBM Model with Linear Kernel

AUC of 0.8555 with bagging technique and an AUC of 0.8545 with boosting technique. Both methods gave better results than the original decision tree model.

4 Results

Most of our models achieved over 85% AUC score, and our best 3 models, SVM with linear kernel, logistic regression, and GBM, achieved AUC about 86%.

From these three models, we consistently observe PC1, same louvain community, PC4, and same greedy modularity community to be our best features, which are two numerical and two categorical predictors.

5 Discussion

First, based on our best two models, logistic regression and SVM with linear kernel, we believe our data likely presents linear boundary.

Second, while our models effectively predicted links for our data, there are some limitations:

- Because of the large data size, we were unable to compute many features, such as betweenness centrality and PageRank.
- We only split the time into two windows, before and after, so it's uncertain how well our model will perform with less data that spans a shorter time period.

For future work, we would like to explore more models and further tune our current models. We also want to employ smaller time-windows to explore granular temporal evolution features. If possible, we would like to learn about the tie-decay algorithm and perform temporal analysis in continuous time.

6 Group Contributions Statement

Jaya conducted exploratory data analysis for the data, performed standardization and PCA transformation of data, and fitted logistic regression, random forest models with 100/200 trees and with max depth = 10/20, LDA & QDA. Chris is responsible for data preparation, which is adding fake edges and features computation, and applying some model performance improvement technique to get a better result. Chris also performed some machine learning models on his own, including logistic regression, SVM, decision tree, random forest, gaussian naive bayes, gradient boosting classifier, etc. Jiaqi Li conducted research on related topics, collected resources for project preparation, performed exploratory data analysis, and worked on improving model performance through feature selection and tuning.

7 Acknowledgement

This report is part of the course project for Math 168 at UCLA. We want to express our appreciation for Prof. Phil Chodrow and TA Grace Li for their guidance and help throughout the course.

References

[1] Walid Ahmad, Mason A Porter, and Mariano Beguerisse-Díaz. Tie-decay networks in continuous time and eigenvector-based centralities. *IEEE Transactions on Network Science and Engineering*, 8(2):1759–1771, 2021.

[2] Michael Fire, Lena Tenenboim-Chekina, Rami Puzis, Ofrit Lesser, Lior Rokach, and Yuval Elovici. Computationally efficient link prediction in a variety of social networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(1):1–25, 2014.

[3] Kyle Griswold, Seokho Hong, and Naveen Arivazhagan. Joint link prediction in temporal networks.

[4] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[5] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[6] Jiali Lin, Zhenyu Li, Dong Wang, Kavé Salamatian, and Gaogang Xie. Analysis and comparison of interaction patterns in online social network and social media. In *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, pages 1–7. IEEE, 2012.

[7] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42, 2007.

[8] Sunil Ray. How to increase accuracy of machine learning model. `https://www.analyticsvidhya.com/blog/2015/12/improve-machine-learning-results/`, Jun 2020.

[9] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 37–42, 2009.

[10] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 981–990, 2010.