Link Prediction for Who-Trust-Whom Online Social Network Using Supervised Learning

Simla Burcu HARMA
Computer Eng. Department
TOBB ETU
Ankara, Turkey
simlaharma@gmail.com

Eşref Oğuzhan Yıldırım
Computer Eng. Department
TOBB ETU
Ankara, Turkey
e.oguzhan.yildirim@gmail.com

Abstract

For social networks, link prediction includes prediction of missing links in a current network or new, also prediction of future links to be built. Link prediction in social networks has taken a considerable amount of attention from both academic and industrial researchers, especially in our era. Before Statistical Learning become so popular, Score-based Approaches were being used in such problems. In recent years the attention to Machine Learning has increased and it occurs that it is more useful than the traditional approach. We will examine Machine Learning approach on link prediction in social networks in this project. We use a dataset[1] including a who-trust-whom social network. In order to do feature exraction we generate some properties of the network graph(common neighbors, Jaccard coefficient, etc.).We apply several Machine Learning methods(Logistic Regression, Decision Trees, Naive Bayes, Stochastic Gradient Descent Classifier, etc.).

1 Introduction

A social network is a social structure consisting a set of social inviduals and their interactions. In order to notate a social network, one can use graphs, nodes representing the people and edges representing the interactions. In recent years, social networks like Facebook, Twitter and Instagram have become considerably popular, so efficient and accurate link prediction algorithms have become a requirement for friend recommendation[2].

Several different algorithms using several different features of social networks have been proposed for link prediction. In our project, we use Supervised Learning and seven features of the given network. Supervised learning in this problem means learning a binary classifier that will predict whether a link exists between a given pair of nodes or not[4]. A similar work has done by Ahmed et al.[3] on Twitter social network dataset. We use a who-trust-whom social network dataset[1] including 75879 nodes and 508837 edges representing people and trust-relationship.

Local and global features of social networks could be used in the link prediction algorithms. Some of the algorithms used in social networks that depend on local features are Common Neighbors (CN), Adamic/Adar (AA) and Jaccard Coefficient (JC). Examples of algorithms that depend on global features are Katz and SimRank. Both types of algorithms are reviewed in [5]. Algorithms that depend on only local features are efficient, but there are global features of the network which may increase the accuracy of the prediction if taken into consideration, like the shortest path between two users, they also ignore recommendations for friends who are not in the neighborhood of the user. Some algorithms, like FriendTNS[6], combine both features.

In our project we extract the following features whose details are given in Section 2: Similarity Metrics, Common Neighbors, Preferential Attachment, Cosine, Jaccard Coefficient, Sum of Neighbors, Average Neighbor Degree Sum.

2 Methods

We are given a dataset including only nodes and edges, thus we need to extract the features in order to train and test the implemented models. We notate the dataset as a directed graph((a, b) is an edge of the graph iff a trusts b).We use the following similarity metrics in this social network graph in order to do feature extraction:

2.1 Similarity Metrics

2.1.1 Adamic Adar Similarity

Adamic/Adar similarity refines the simple counting of neighbour's trustworthiness by weighting trusted neighbours more heavily.

$$AA(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log(|\Gamma(z)|}$$

2.1.2 Common Neighbors

Common neighbors predictor declare us that two stranger -neither of them trusts the other- people who have a common neighbor -either A trusts B or B trusts A for people A and B- may trust each other in the future.

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

Newman [8] has come to a conclusion after his experiment on collaboration networks that if strangers A and B has sufficient number of neighbours then after some time t, A and B will probably collaborate.

2.1.3 Preferential Attachment

One well-known concept in trust networks is that people with many neighbours tend to create more trust relations in the future. This is in fact in some social networks, like popularity, popular person become more popular. We score the trustworthiness of two people by this property with multiplying their number of neighbours.

$$PA(x, y) = |\Gamma(x)| * |\Gamma(y)|$$

2.1.4 Cosine

This predictor punishes preferential attachment but considers the common number of neighbours as well.

$$CS(x, y) = \frac{CN(x, y)}{PA(x, y)}$$

2.1.5 Jaccard Coefficient

The Jaccard coefficient measures the ratio of number of common neighbors of A and B to number of total neighbors of A and B. It is, intuitively a criterion for two people trusting each other.

$$JC(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

2.1.6 Sum of Neighbors

Sum of neighbours predictor tells us that if person A has more neighbours than B, it is likely to be a stranger C trusts A before it trusts B. This property is used in social friendship networks but also our projects showed us that it has a significant role on trust networks.

$$SN(x, y) = |\Gamma(x)| + |\Gamma(y)|$$

2.1.7 Average Neighbor Degree Sum

Average neighbour degree sum, as can be understood from its name, uses two peoples neighbours and their trustworthiness. It averages people's neighbours' trustworthiness(degrees).

We coded all these metrics in Python genericly.

Furthermore we generate the training and test data from the given social network. In terms of the definition of the problem, we will be given a pair of nodes and we will try to predict whether the first node(person) trusts the second, we consider trust relationship as positive examples(class:1) and no-trust relationship as negative examples(class:0). In order to generate the training data, we take $4/5$ of the given network. Since the data set provided contains only trust relationship we have $508837 * (4/5) = 407069$ positive examples at first. We add 407069 negative examples -obtained randomly from all possible edges that the graph does not contain- to balance the data. Consequently we obtain 814138 train node pairs and extract their features by considering this as a seperate network. Then, we take all the data set, do the same process-adding negative examples-, and extract their features, and use the output matrix containing 1017674 pairs as the test data.

After extracting the features, we normalize them by coding in Python. Finally, we print both train and test data that we obtained, to the separate files.We use these files for training and testing the models with Weka which is explained below.

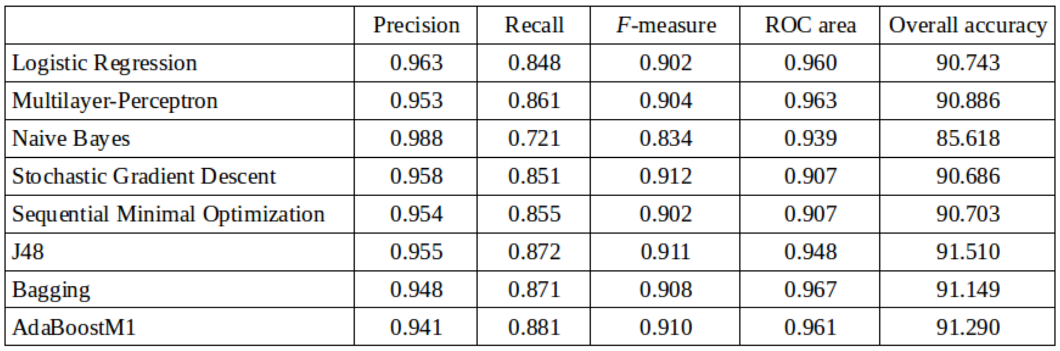
Weka is a useful software collecting several machine learning algorithms for data mining tasks[7]. We apply following classification methods in Weka:

Classification Methods

1. Logistic Regression
2. Multi Layer Perceptron
3. Naive Bayes
4. Stochastic Gradient Descent
5. Sequential Minimal Optimization
6. C4.5 Decision Trees(J48)
7. AdaBoostM1

3 Experimental Results

As we can see from the tables and line chart above, eight models are used for dataset. We execute all eight models several times and note down their best performance. Naive Bayes model gave the worst result according to others in all metrics. If we compare models as their root relative squared root error, Bagging has least error thus it gives the best performance. If we compare models as their overall accuracy score, J48 has the highest overall score.



	Precision	Recall	*F*-measure	ROC area	Overall accuracy
Logistic Regression	0.963	0.848	0.902	0.960	90.743
Multilayer-Perceptron	0.953	0.861	0.904	0.963	90.886
Naive Bayes	0.988	0.721	0.834	0.939	85.618
Stochastic Gradient Descent	0.958	0.851	0.912	0.907	90.686
Sequential Minimal Optimization	0.954	0.855	0.902	0.907	90.703
J48	0.955	0.872	0.911	0.948	91.510
Bagging	0.948	0.871	0.908	0.967	91.149
AdaBoostM1	0.941	0.881	0.910	0.961	91.290

Figure 1: Detailed Accuracy of the Methods.

	Kappa Statistic	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error	Root relative squared error
Logistic Regression	0.8149	0.1500	0.2691	30.0027%	53.8167%
Multilayer-Perceptron	0.8177	0.1205	0.2631	24.1031%	52.6154%
Naive Bayes	0.7124	0.1435	0.3743	28.7055%	74.8691%
Stochastic Gradient Descent	0.8137	0.0931	0.3052	18.6284%	61.0383%
Sequential Minimal Optimization	0.8141	0.0930	0.3049	18.5930%	60.9803%
J48	0.8302	0.1337	0.2601	26.7422%	52.0218%
Bagging	0.8230	0.1205	0.2588	24.1078%	51.7632%
AdaBoostM1	0.8258	0.1251	0.2663	25.0150%	53.2516%

Figure 2: Evaluation Metrics of the Methods.

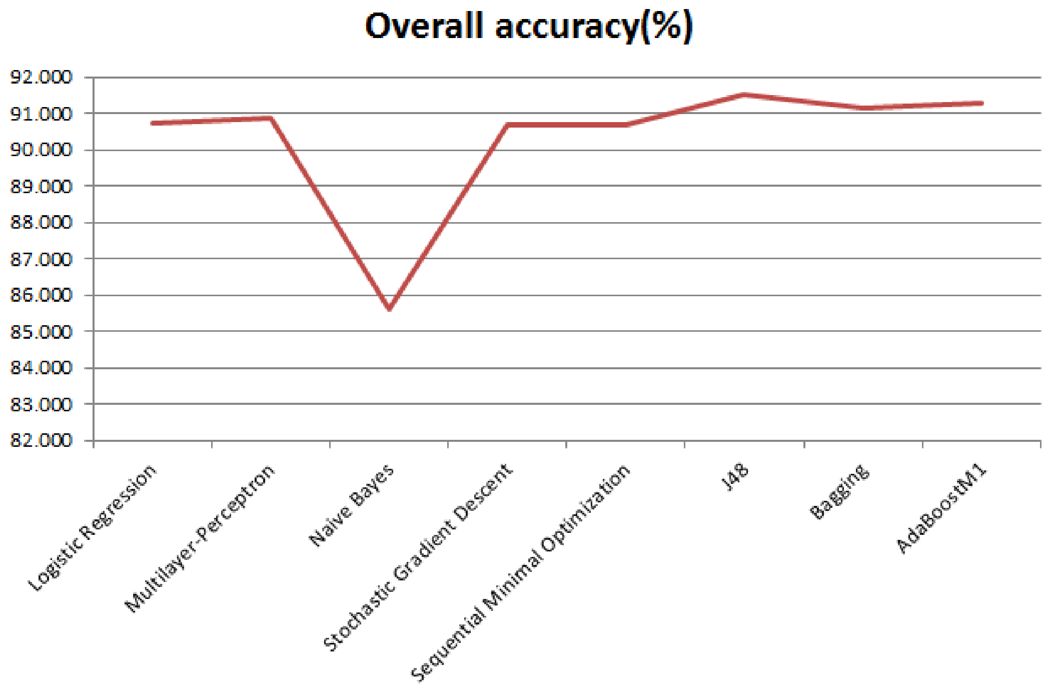


Figure 3: Overall Accuracy of the Methods.

We try to run Random Forest and KNN models, however these models take large amount of time to finish with big datasets like ours. We use two Ensemble Classifiers, Bagging and AdaBoostM1, and six Single Classifiers, others. There is no actual dominancy among Ensemble and Single Classifiers. However both Ensemble Classifiers are in three best models according to all metrics.

In this project, over 1 million data has been used. That means even the small amount of change matters. In terms of overall accuracy, J48 leads the others with 91.510%. Second best model is AdaBoostM1 with 91.290%. Even though difference between these two models may seem trivial, with 1 million examples, this difference(0.220%) causes 2200 incorrect predictions, even more for much larger datasets. So the model selection is the most crucial part for classification accuracy.

4 Conclusion

A supervised link prediction on who-trust-whom social networks framework was introduced using variety of features. Seven features generated with similarity metrics according to the dataset. Training and test data are generated from the given social network. Features are extracted for each training and test pair, separately. After that, eight models were trained and tested several times, and best results for each models considered. Models' evaluation metrics were driven from test results and noted down for detailed comparison. According to overall accuracy metric, J48 Classifier has the highest score with 91.510%.

For future work, we believe that we can add more features to dataset for a better representation of network. In addition to this, better connected network may have effect models positively.

References

[1] https://snap.stanford.edu/data/soc-Epinions1.html.

[2] Aiello L M, Barrat A, Schifanella R, et al. Friendship prediction and homophily in social media. In: ACM Transactions on the Web, 2012, 6: 9.

[3] Ahmed, Cherry, Abeer ElKorany, and Reem Bahgat. A supervised learning approach to link prediction in Twitter. In: Social Network Analysis and Mining 6.1, 2016: 1-11.

[4] M. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In: Proceedings of the Workshop on Link Discovery: Issues, Approaches and Applications, 2005.

[5] Liben-Nowell D, Kleinberg J. The link prediction problem for social networks. In: Proceedings of CIKM'03. ACM Press, 2003. 556–559.

[6] Symeonidis P, Tiakas E, Manolopoulos Y. Transitive node similarity for link prediction in social networks with positive and negative links. In: RecSys, 2010.

[7] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The weka data mining software: an update. In: SIGKDD Explor News, 2009, 11:10–18.

[8] M. E. J. Newman. Clustering and preferential attachment in growing networks. In: Physical Review E, 2001, 64(025102).