Final Report: Link Prediction in the Pinterest Network

Poorna Kumar
poornak@stanford.edu

Amelia Lemionet
lemionet@stanford.edu

Viswajith Venugopal
viswa@stanford.edu

Abstract

Link prediction is a classic problem in networks, of great practical relevance. In this project, we focus on predicting links from boards to pins in the Pinterest network.We begin by doing in-depth exploratory analysis to understand the structure of the Pinterest network, and we investigate its evolution over time, highlighting some important properties along the way. We then build training sets of the data in two different ways, and implement both scoring-based and supervised learning methods to perform link prediction on the data, specifically between links and boards. We motivate and compute a set of relevant features and scoring functions from the network to use in our methods. We find that among supervised learning methods, SVMs are the most robust to unbalanced training data, and that among the scoring functions, Personalized PageRank and common neighbors show promising results.

1 Introduction

In this paper, we focus on performing link prediction on the Pinterest network. Given a network, the link prediction problem is defined as follows. Consider the network at times t and t', G_t and $G_{t'}$, where $t' > t$. The goal is to use G_t to predict the *new links* that will be formed – that is, the pairs of nodes (x, y) which are connected by an edge in $G_{t'}$ but were not connected in G_t. This is an interesting problem – in addition to being quite challenging [1] [3], it also has enormous utility in industry. For example, Facebook and Twitter use it to suggest people to add or follow, Amazon uses it to suggest products for people to purchase and so on.

We use a Pinterest dataset provided by Prof. Leskovec, which is a tri-partite network of users, boards and pins, where we have the timestamps at which different nodes and links were created. The structure of this network is fairly unique among datasets on which network analysis in general, or link prediction in specific, is typically done in the literature – this makes for some fairly interesting findings when understanding the structure and evolution of the network, and presents new challenges as well as possibilities for link prediction. Our insights and methods will be useful in helping Pinterest build a recommendation system – specifically, one that recommends to the owner of a board, some pins that they can add to it.

The rest of this report is organized as follows. Section 2 presents a brief review of related work in the literature. In Section 3, we explore the properties of our network, and describe our insights into the structure and evolution of this network. In Section 4, we discuss methods for subsetting our data to split our graph into training and test edges, and we outline some supervised and unsupervised methods for link prediction. In Section 5, we describe our evaluation metrics for these methods, and then analyze our results. Finally, we dwell on the caveats of our analysis, and touch briefly upon directions for future work in Section 6.

2 Related Work

2.1 Link Prediction based on Similarity Scores

One approach for link prediction is to assign similarity scores to pairs of nodes which are not currently connected, and use these scores as a measure of how likely the nodes are to be connected in the future. We can thus produce a ranked list of node pairs in decreasing order of scores. Liben-Nowell et al. [9] provides an overview of similarity measurements (both local and global) that can be used. Among others, these methods include shortest-distance-based recommendations, number of common neighbors, and Personalized PageRank, all of which we adapt to solve our problem, as described later in the report.

2.2 Link Prediction as a Supervised Learning Problem

A significant body of work [1] [3] deals with the formulation of link prediction as a supervised machine learning problem. Using the same snapshots G_t and $G_{t'}$ as before, we can generate a training set as follows: we choose pairs of nodes (x, y) which are not connected by an edge in G_t as examples; the label is positive if the nodes are connected in $G_{t'}$, and negative if they are not. For each example, we can compute features based on x, y and G_t – these are similar to the similarity scores discussed in the previous subsection, which can be based on topological attributes, distance metrics or neighborhood-based metrics computed from the network, as well as node and edge attributes.

Al Hasan et al. [1] uses this formulation to perform link prediction on academic co-authorship networks. They generate the training set labels as described above, and a combination of several features as input. They then use several machine learning classification models, like decision trees [13], SVMs [14] and Naive Bayes [8], and discuss the performances of the different models.

2.3 Link Prediction and Network Analysis of Bipartite Networks

There are many key differences between bipartite and one-mode networks, some of which are discussed in [7] and [4] (in the context of clustering, but these challenges apply to the link prediction problem as well). For example, bipartite networks do not display the property of triadic closure, unlike most human social networks. Benchettara et al. [3] apply supervised learning to predict links in a bipartite network, by proposing specific additional features to use in bipartite networks. These features are computed by considering one-mode projections of the bipartite graph.

2.4 Other Methods

There are several other methods for link prediction that have been proposed, and shown to work well on real datasets. [10] talks about the use of local random walks as a way of performing link prediction on large graphs, which would be more computationally expensive with other methods. [2] takes this a step further, proposing a method called Supervised Random Walks which formulates link prediction as an optimization problem, so as to combine topological features computed from the network with node and edge attributes when predicting links.

3 Exploring the Data and Analyzing the Evolution of the Network

3.1 Overview of the dataset

Pinterest is a social networking site which bills itself as a 'content sharing service'. It allows users to create 'boards', which are collections of related 'pins'. Pins can be images, videos or other objects from the internet or from other boards on Pinterest, which users can add to their boards. Users can also follow boards from all over Pinterest that they're interested in.

The dataset we use consists of the food-themed boards in Pinterest, the users who created them, the pins in them, and the users who followed them, all from 2010 to 2014. The user-to-board graph is a bipartite graph, and the board-to-pin graph is another bipartite graph. By stacking these two graphs, we get a tripartite graph, visualized in Figure 1. Since Pinterest was created in 2010, the number of users on the platform (and thus of boards and pins) has increased substantially over time. Our entire dataset has 12.5 million boards, 20 million pins and 18.5 million users. In the following two sections, we present descriptive analyses of the user-to-board and board-to-pin bipartite graphs.

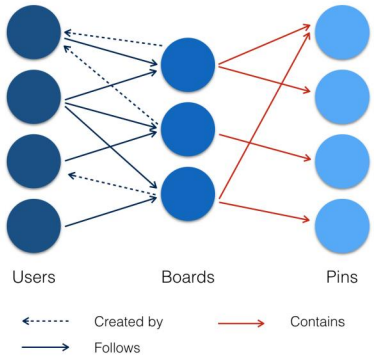


Figure 1: Illustration of Pinterest tripartite network

3.1.1 User-board bipartite graph

There are two kinds of links between users and boards: 'follow' links for when a user follows a board, and 'create' links between a board and the user who created it. Of our 18.5 million users, 14 million of them follow at least one board; 7 million of the 12.5 million boards are followed by at least one user.

On average, each user follows 3.5 boards, but the median is only at 1. On the other hand, the mean board has 6.3 followers, and the median board has 2. As for the number of 'follows', it has been increasing at a relatively constant rate since the

beginning of 2012 – there are no follow links in our dataset prior to January 2012, and we believe this probably corresponds to the introduction of the follow feature.

Looking at the board creation links, we find that the mean number of boards created per user in our dataset is 1.2 (which is very close to the median of 1.0). In the early days of Pinterest, up to the end of 2011, the rate at which boards were being created was quite low – after this, however, it rapidly ramps up, with many more boards being created per day.

3.1.2 Board-pin bipartite graph

The degree histograms of boards and pins in the pin-board bipartite graph are plotted in Figures 2 and 3.

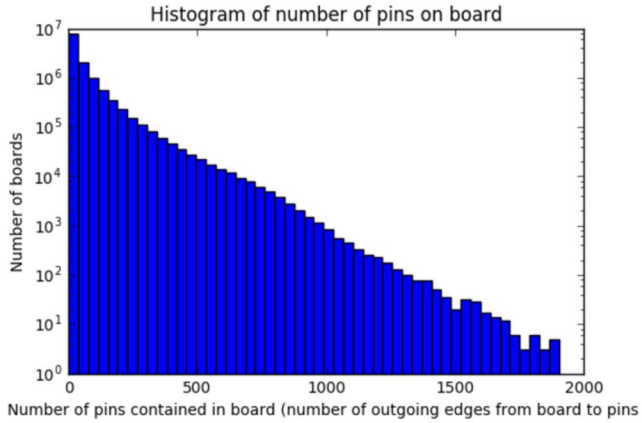


Figure 2: Degree distribution of boards (log scale on y-axis)

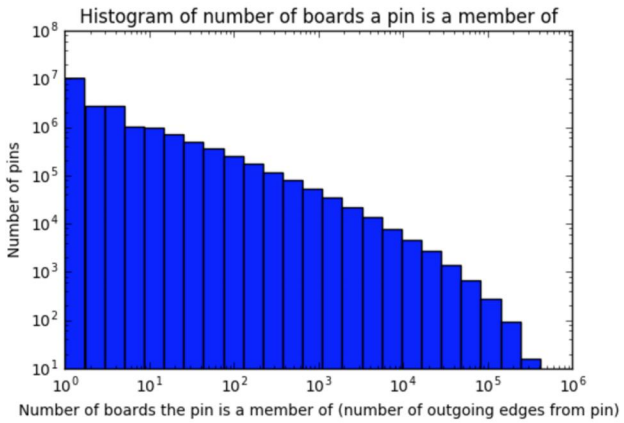


Figure 3: Degree distribution of pins (log-log scale)

The bipartite pin-board graph contains 736 million board-pin links. Most boards contain fewer than a hundred pins, and the median number of pins per board is 26. However, there are a few large boards, the biggest of which contains 1904 pins.

As for pins, more than half the pins appear only on one board. We suspect that this correspond to pins that a user stumbles on on the internet, but are never picked up by any other user on to their board. This indicates that a lot of edges are created when pins come into the network from outside the network, and these edges are impossible to predict without a knowledge of the user's browsing patterns and the structure of the internet itself.

Finally, note that although around half the pins appear only on one board, we also have some viral pins, with the most popular pin appearing on over 400,000 boards. These are uncommon, though, and 85% of pins appear on just ten boards or fewer.

3.2 Structural Evolution of the Network Over Time: An Analog to Triadic Closure

Triadic closure is a commonly observed property in real-world social networks, i.e., if A and B are linked, and B and C are linked, then A and C have an increased probability of being linked at some point in the future[11]. But in a bipartite graph such as ours, triadic closure does not directly hold, since we have no triangles. In this section, we develop and test an analogous hypothesis for our graph, motivated by our reasoning about how Pinterest works.

First, we present the intuition behind our hypothesis. Since users on Pinterest can 'follow' boards, they are exposed to a feed of content (pins) that is partly curated from the boards that they follow. Additionally, if a user follows a board, it is likely that she/he is especially interested in the content of the board. Since a user is likely to be both interested in and exposed to the content on boards that they follow, we believe that there is a higher probability that, when the user is adding pins to their own board, they add pins which are present on the boards that they follow (in comparison to pins that are not present on any board which they follow).

To make this hypothesis concrete, consider a user u_k who follows m_t boards at time t. Let these boards be denoted by b_i, where $i \in \{1, ..., m_t\}$. Also, let user u_k create board b_{u_k} at some point before time t. Now suppose that, at time t, the user adds a pin p to board b_{u_k}. We wish to investigate the probability that pin p is contained on one of the boards b_i, $i \in \{1, ..., m_t\}$, and contrast this to a null (random) graph model.

3.2.1 Null Model Specification

First, consider the case that p has only one other outlink to the network (i.e., it is contained on only one other board), before it is added to the board b_{u_k}. In our null random graph model, we assume that at time t, given that a board b_i has $d_{i,t}$ outlinks to pins, and given that our pin of interest p has one outlink to some board, the probability that one of the outlinks of b_i is to pin p is $b_{i,t}/T_{pbl,t}$, where $T_{pbl,t}$ is the total number of pin-board links in the graph at that time. In other words, if we know that a pin has one outlink, we assume that any outgoing edge from any board is equally likely to meet this outgoing link from the pin, and form a connection. So, the probability that p is contained on *any* of the boards b_i, $i \in \{1, ..., m_t\}$ is

$$\Pr(\text{Any board } b_i \text{ links to } p \text{ at } t) = \sum_{i=1}^{m_t} \Pr(b_i \text{ contains the pin at time } t) = \sum_{i=1}^{m_t} \frac{d_{i,t}}{T_{pbl,t}} \tag{1}$$

$$\text{Then, } \Pr(\text{No edge from } p \text{ to any } b_i \text{ at } t) = 1 - \sum_{i=1}^{m_t} \frac{d_{i,t}}{T_{pbl,t}} \tag{2}$$

We now consider the case where p has multiple links to the network before it is added to board b_{u_k}. Let the number of such edges be $d_{p,t}$. We construct an expression for this probability with $d_{p,t} = 2$, and then extend it to arbitrary $d_{p,t}$. When $d_{p,t} = 2$, the probability that *neither* of the two outlinks connects to any of the boards b_i, $i \in \{1, ..., m_t\}$ at time t is given by

$$\begin{aligned} \Pr(\text{No edge from } p \text{ to any } b_i \text{ at } t) &= \Pr(1^{\text{st}} \text{ edge does not go to any } b_i) \times \\ &\quad \Pr(2^{\text{nd}} \text{ edge does not go to any } b_i | 1^{\text{st}} \text{ edge does not go to any } b_i) \end{aligned} \tag{3}$$

$$\approx \left(1 - \sum_{i=1}^{m_t} \frac{d_{i,t}}{T_{pbl,t}}\right)\left(1 - \sum_{i=1}^{m_t} \frac{d_{i,t}}{T_{pbl,t}}\right) \tag{4}$$

where the approximation comes from the fact that if we are conditioning on the first link from p already being formed, there are fewer outgoing edges from boards that the second link has the option to meet (since a pin cannot be pinned twice on the same board). However, since the decrease in the number of options for the second link is at most the degree of any board (and on average, much less than the maximum degree of any board in the graph), and because this number is much lower than the total number of pin-board links in the graph at time t, we can ignore this decrease.

We can extend this to arbitrary $d_{p,t}$, if $d_{p,t} \ll T_{pbl,t}$ (which is true). Under our null model, when pin p is added to board u_k, we have

$$\Pr(\text{Edge exists from } p \text{ to } b_i) = 1 - \Pr(\text{No edge from } p \text{ to any } b_i) = 1 - \left(1 - \frac{\sum_{i=1}^{m_t} b_{i,t}}{T_{pbl,t}}\right)^{d_{p,t}} \tag{5}$$

Figure 4: Number of board linked to pins contained on boards followed by user

3.2.2 Methods and Results

With this null model in mind, we track the empirical evolution of our network over time. For any pin-board link that is formed between any user u_k's board b_{u_k} and p, we can find, under our null model, the probability that the pin p is on a board b_i, $i \in \{1, ..., m_t\}$, that the user u_k follows, and denote this probability by $p^{\text{null}}_{\text{link}}$ (which is a function of time t, user u_k, pin p, etc., i.e., this probability is different for different links in the network). We can check whether the newly added p

was indeed on some other board that user u_k follows, i.e., we have an observation of what really occurred in the evolution of our network.

In our graph, we summed $p_{\text{link}}^{\text{null}}$ over all pin-board links formed in the network over time, to get the expected number of occurrences of this phenomenon. We then contrasted the expected value with the observed value. Figure 4 plots the observed and expected results for each month from January 2012 (when the 'follow' feature first appears in our data) to the end of 2014. We find that the actual number of occurrences of this phenomenon is much higher than the expected number (for most months, the ratio exceeds of observed value to expected value exceeds 2) i.e., there is a significant increase in the probability of adding a pin to one's board if that pin exists on boards that one follows.

We had previously touched upon the fact that a lot of boards on Pinterest are discovered through the wider internet, and that a large fraction of these links are basically unpredictable (since 50% of the pins in our network have a degree of just 1). This discovery signals a degree of predictability in the other kinds of links that are formed, and highlights a property of our network that is analogous to triadic closure: if a pin is on a board that a user follows, then it is more likely to be added to the user's boards than a pin that is not. We use this insight in picking features for our link prediction algorithms.

3.3 Detecting Virality Early

In score-based unsupervised learning methods, when recommending a pin to a board, we rank all the pins in the network and then predict links to the top-ranked pins. The number of pins on Pinterest makes this a computationally expensive task. Further, we have seen that most pins in the network are of degree 1, i.e., they never propagate beyond the initial board they were placed on. In this situation, some pruning of the network is called for.

In the real-world recommendation problem, it is important to tell in advance which pins are going to be popular – else, we might throw them out of our network while pruning because they have low degree, even though they are rapidly gaining popularity and will soon have very high degree. How to detect virality early and accurately is an active subject of research [15] [6] and can involve sophisticated methods, but we find some simple heuristics which can be used as a first pass to identify these pins.

In particular, we find a strong positive relationship between the number of edges a pin forms in the first month of its life on the network, and the number of edges it forms after. We test this hypothesis by looking at all pins created in 1 day (1^{st} June 2014) in the network (so as to control for the other properties of the network, such as network structure, seasonal trends, etc.). We track the number of links added to each of these pins over the month of June 2014 (N_1), and also how many links are added to these pins *after* June 2014 (N_2). We find that the number of edges a pin adds in the first month of its existence is significantly correlated with how many edges it gains after the first month (a Chi-Square contingency test on these results gave us a p value of 2.3e-6). We performed the same analyses on other days, and obtained similar results.

Table 1: Relationship between number of pins added in the first month and after

	$N_1 < 100$	$N_1 \geq 100$
$N_2 < 1000$	34373	172
$N_2 > 1000$	9	43

4 Methodology

4.1 Score-based Recommendation Methods

As described in Section 2.1, score-based methods recommend pins to a board by using scoring functions, which assign a score to each pin that the board is not connected to in G_t. The pins with the highest scores are then presented as recommendations. The scoring functions we used are:

- **Personalized PageRank:** [12] This is a powerful measure of proximity between nodes in a network. To find the Personalized PageRank score of all the pins in the network with respect to a board b, we perform a random walk starting from the board b, and at every step, with a small probability (0.15 in our implementation), we teleport back to b. The Personalized PageRank score of each of the other nodes is the fraction of time steps the random walk will spend at that node. In practice, this score can be more efficiently computed as the stationary point of a matrix power equation – although the time for computation is still prohibitive for large networks such as ours.
- **Common neighbors:** We tweak the notion of common neighbors to suit our bipartite graph. Specifically, given a board b, we compute a score $S'_b(p_i, p_j)$ for every pair of pins p_i, p_j in our network such that p_i is pinned to b and

p_j is not pinned to b in G_t:

$$S'_b(p_i, p_j) = \frac{|\text{common neighbors between } p_i \text{ and } p_j|}{|\text{neighbors of } p_i|} \tag{6}$$

With this, the score of a candidate pin p for the board b is:

$$S(b, p) = \sum_{p_i \text{pinned to } b} S'_b(p_i, p). \tag{7}$$

- **(Negated) Shortest Path Length:** We find the length of the shortest path between two nodes, and use that as a measure of distance between them (and thus, negating this gives us a similarity measure).
- **Product of Degrees:** In keeping with the rich-get-richer (preferential attachment) intuition, we use the product of node degrees as a measure of similarity between two nodes (thus expressing the intuition that two nodes of high degree are likely to form edges with each other).

4.2 Supervised Learning Methods

As described in Section 2.2, board to pin link prediction can be formulated as a supervised learning problem. Given a board and a pin (b, p) which are not linked in G_t, we compute a set of features for this pair. Our response variable denotes whether the pair is connected in the future or not.

Our features can be grouped in four broad categories: those largely related to the board, those largely related to the pin, those related to the user who created the board, and finally those that depend on the specific relationship between the user, pins and board. These features are described below.

Category	Feature
Board information (b_i)	Number of pins in b_i Time since b_i was created Time since a pin was last added to b_i
Pin information (p_j)	Number of boards p_j belongs to Total number of followers of the boards p_j belongs to (popularity) Time since p_j was last added to a board
User information (u_k, creator of b_i)	Average number of pins per board created by u_k Time since u_k's last activity (board creation or pin addition)
Boards, pins and users	Number of boards followed by user u_k on which p_j is pinned Number of boards p_j co-occurs with other pins from board b_i Number of boards created by user u_k that contain pin p_j

We train SVMs (with a RBF kernel), Random Forests and logistic regression classifiers on our training data, and evaluate these methods by predicting the labels on our test dataset. In short, SVMs find a separating hyper-plane between positive and negative training examples, by mapping the features to a high-dimensional space. Random Forests are an ensemble learning method of bagged decision trees, where the number of decision trees is a hyperparameter to be specified. Logistic regression gives us a probability of a sample being positive or negatively labeled, by modeling the log-odds function as a straightforward regression.[5]

4.3 Generating Train and Test Graphs

We generate two different datasets from the Pinterest network – in one, we use the network as it is, and in another, we take a 'hyperactive' subgraph of Pinterest, retaining only pins and boards with high degrees, i.e., 'hyperactive nodes'. We reason that these pins and boards are the ones that are most likely to form links in the network.

4.3.1 Unpruned Network

For the unpruned network, we took all the data from the beginning of our dataset up to June 1st 2012 as our snapshot, G_t. This graph had 9.2 million nodes and 64 million edges. Although this network is huge, we chose these dates because we wanted to make sure our network had enough 'follow' links, which were only introduced in January 2012. The links of interest that we tried to predict were those that were formed between June 1st 2012 and July 1st 2012.

4.3.2 Hyperactive Subgraph

We also generated a network consisting of the 'hyperactive' subgraph of Pinterest – we retained only the boards with degree at least 500, and pins with degree at least 5000. In this network, we took the snapshot of the graph as it was on Jan st 2013 as G_t, and tried to predict links formed from Jan 1st 2013 to July 1st 2013. This graph had 100,000 nodes in total.

4.3.3 Generating Datasets for Supervised Learning

In the two networks we generated, we define 'present' as the edges which exist in G_t, and the 'future' as edges which exist in $G_{t'}$ but not in G_t. To generate our dataset, following the work from [1], we randomly select board-pin pairs (b, p) which are not linked in the 'present'. If a pair (b, p) is linked in the 'future', we assign it the label 1, and if it is not linked in the 'future', we assign it the label -1. Finally, we split the resulting dataset into training, validation and test sets.

Sampling board-pin pairs uniformly at random would result in a very-skewed distribution of positive and negative examples in a network as large and sparse as ours, which has only 1 edge per every million possible board-pin pairs. In order for most supervised learning methods to work, we need a training dataset which has a fairly balanced distribution of positive and negative observations. If we sampled board-pin pairs uniformly at random, we would have a dataset that would be too large to handle, and our model would have very low precision when predicting positive links (since it would essentially always predict -1). Thus, we deliberately introduce pairs which are in the 'future' to generate a dataset that had an even distribution of positive and negative examples.

5 Experiments and Results

5.1 Scoring-based methods

Evaluation Metrics

We evaluated the scoring functions described in 2.1 on the 'hyperactive' subgraph of Pinterest by first taking snapshots of the subgraph at times t and t'. Then, we sample boards which are in the network at time t, and score all the pins that are *not* connected to the board at time t, using the scoring functions we discussed. We then rank all the pins in decreasing order of this score.

For consistency, we normalize the ranks by dividing the rank of each pin by the total number of pins in the rank list (which is exactly the number of pins not connected to the board at time t). Thus the normalized rank varies between 0 and 1, with higher ranked pins having values closer to 0. Then, for all the pins that were, in the real network, pinned to the board between t and t', we look at what their normalized rank was according to our scoring function. If our recommendations were perfect, the pins who were actually added to the board would be ranked highest – that is, their normalized ranks will be numerically very low.

In keeping with this intuition, we use the following evaluation metrics. We compute the mean normalized rank of the pins that were actually added to our sampled boards between t and t'. We also compute the fraction of pins with normalized rank < 0.1, which translates into pins which were ranked in the top 10% for a given board. Second, we look at the distribution of the normalized ranks of the pins, and compare it to the random baseline. Under the random baseline, all rankings of pins are equally likely, and therefore the normalized ranks of the pins actually added to the boards would be distributed uniformly over pins.

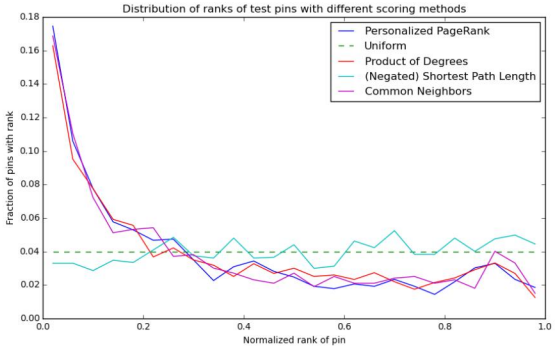


Figure 5: Distribution of normalized ranks of pins added between t and t' using different scoring methods

Results

The following table presents our results, and Figure 5 shows the distribution of the normalized ranks of the test pins. The x axis represents all the possible values for normalized rank, and the y axis represents the fraction of pins with any given rank value, averaged over all boards.

The performance of the uniform random baseline is shown using a dashed line. We can see that Personalized PageRank, degree product and common neighbors perform very similarly, and they all beat the random baseline soundly –

Table 2: Results of scoring-based methods

Scoring method	Mean normalized rank	Fraction of pins with normalized rank < 0.1
Personalized PageRank	0.328	0.323
Degree Product	0.344	0.299
Common Neighbors	0.339	0.320
Shortest Path Length	0.526	0.080
Random Baseline	0.500	0.100

however, they are far worse than a perfect scoring function would be, which confirms our intuition that link prediction is difficult. Further, the shortest path length-based scoring does not even outperform the random baseline.

5.2 Supervised Learning

Evaluation Metrics

We compared SVM, Logistic Regression and Random Forests based on three measurements: accuracy, F1 score and AUC.

Accuracy is the fraction of observations from the test set that are correctly labeled by a classification method. In general, accuracy is a good reflection of how well a prediction method performs whenever both the classes are equally represented in both the training and testing set (if this is not the case, any classifier the predicts the majority class would get a high accuracy, but would be a sub-par predictor for the minority class). Here, we used accuracy when predicting on the training and test sets described in section 4.3.1 where both classes are equally represented.

The F1 score is a more robust measurement that generalizes to sets where classes are represented disproportionately, since it takes into account both precision and recall. In this paper, we chose to use the F1 score of the positive class, since this class is the hardest one to predict in our sparse network.

Finally, an ROC curve represents the trade off between true positive and false positive rates, and the area under the ROC curve, known as AUC, is a measurement of the performance of a classifier. Because it takes into account true positive and false positive rates, AUC is more robust than accuracy to disproportionately distributed response variables. We report AUC for our different classifiers. Further, we also used AUC as our criterion for variable selection by running backward elimination with SVM.

Results on unpruned subgraph

We tested the supervised learning methods on the dataset described in section 4.3.1, where both the training and test set were built such that they contained 50% of the observations to be positive and 50% to be negative.

Prediction method	Accuracy	AUC	F1-score (class 1)
SVM	0.9129	0.9129	0.9136
Logistic regression	0.9227	0.9227	0.9228
Random Forests[1]	0.7570	0.7569	0.7947

Noticeably, all three methods perform well on our test set by any of our evaluation metrics, although random forests underperform compared to the other two methods. In fact, if the Pinterest network was such that every link had a probability 0.5 of being formed, these supervised learning methods would be very good link predictors. The actual structure of the Pinterest network has a much more unbalanced proportion of positive and negative examples, close to one positive example for every 10^6 negative ones. In further paragraphs we will analyze how these methods would perform with more sparse networks. We were also interested in understanding the predictive power of our different features, in the case where our test set was well-balanced. We performed backward elimination using an SVM, and found that the five most relevant variables in terms of AUC for predicting edges between pins and boards were, in order: (1) time since the pin was last added to a board, (2) number of pins in the board, (3) popularity of the boards where the pin belongs to, (4) number of boards the pin belongs to, (5) time since the board was created.

It is important to note that after adding two variables to the model the AUC value flatted out somewhat through the addition of more variables. We reason that this is because most of the predictors we are using are highly correlated.

Finally, in order to understand how the previous results scale when predicting on a test set where the number of positive examples is small compared to the number of negative ones, we tested the models (and computed their corresponding

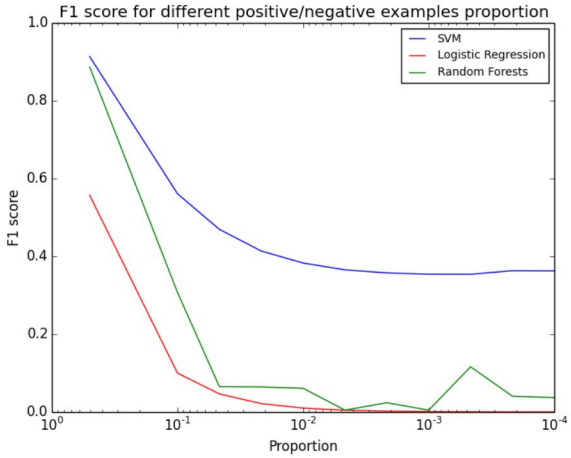


Figure 6: F1-score for changing structure of the test set

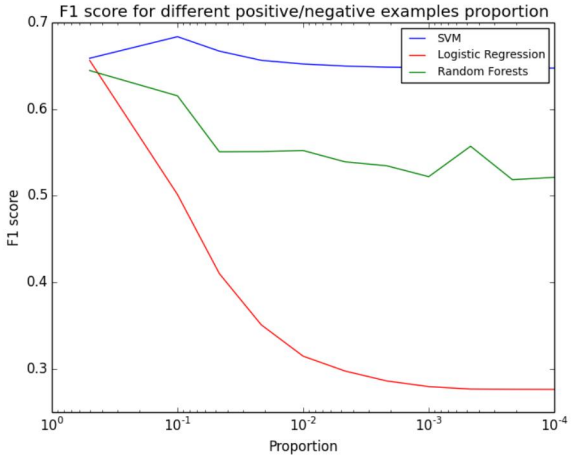
F1-scores) for different proportions of positive to negative examples in the test set, while always maintaining the same training set (with a 50-50 proportion). The results are shown in Figure 6

SVM seems to be the most robust method when changing the proportion of positive to negative examples. As the plot shows, when this proportion reaches 10^{-2}, the F1-score stabilizes around 0.4, even as the proportion of positive examples continues to decrease. Therefore, out of the three supervised methods tested on this section, SVM appears to be the best suited for the structure of the Pinterest data.

Results on Hyperactive Sub-graph
To focus solely on a hyperactive section of Pinterest, we tested our methods in a subset of the data 4.3.1 where we only consider pins that have a degree of at least 5000. We used the same features as before.

Prediction method	Accuracy	AUC	F1-score (class 1)
SVM	0.6562	0.6587	0.5919
Logistic regression	0.6521	0.65490	0.5811
Random Forests[2]	0.6465	0.6465	0.65137

We notice here that the performance of the predictors is lower than what we observed for the previous data set, probably because when all nodes in the subgraph are hyperactive, we have that positive and negative examples looking more similar on average. Nonetheless, when we change the proportion of positive examples in the test set, we observe a higher F1-score for more sparse datasets compared to what we observed in the previous section. This indicates that our methods will more accurately predict links for the actual Pinterest structure for hyperactive nodes.



In terms of variable relevance, based on AUC with SVM, the three most relevant variables proved to be: (1) number of boards the pin belongs to, (2) time since the last pin was added to the board and (3) time since pin was last added to a board.

6 Conclusions and Future Work

In this report, we performed network analysis to understand the structure and evolution of the Pinterest network, and implemented various methods to predict board-pin links in the network. We showed that a significant fraction of pins that are added by users are from boards that they follow. We also showed that virality of a pin could be detected early based on the number of links it forms in the first 30 days of existence.

We generate two different datasets from the datset, one unpruned and one with only 'hyperactive' nodes. We used scoring functions for link prediction, and found that Personalized PageRank and common neighbors performed well. The supervised learning methods we implemented performed well on an evenly split test set; as we skewed the test set further in order to be closer to the true proportions, we found that only SVMs retained a respectable F-score. We found that the best predictors of link formation were pin degree, and how long it had been since the pin and board were last active.

There are several promising directions for future work. These include: performing a deeper analysis to understand virality in this network, so as to find more principled ways of pruning the dataset (to simulate the real-world link prediction problem); finding ways to make our prediction functions more computationally efficient and scalable; trying to predict user-board follow links in addition to pin-board links; and using other metadata (such as information about the board and the user) to aid in our predictions.

References

[1] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning.

[2] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[3] N. Benchettara, R. Kanawati, and C. Rouveirol. Supervised machine learning applied to link prediction in bipartite social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 326–330. IEEE, 2010.

[4] S. P. Borgatti and M. G. Everett. Network analysis of 2-mode data. *Social networks*, 19(3):243–269, 1997.

[5] T. Hastie and Friedman. The elements of statistical learning.

[6] P. Jain, J. Manweiler, A. Acharya, and R. R. Choudhury. Scalable social analytics for live viral event prediction. 2014.

[7] J. Kunegis. Exploiting the structure of bipartite graphs for algebraic and spectral graph theory applications. *Internet Mathematics*, 11(3):201–321, 2015.

[8] D. D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer, 1998.

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

[10] W. Liu and L. Lü. Link prediction based on local random walk. *EPL (Europhysics Letters)*, 89(5):58007, 2010.

[11] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[12] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[13] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. 1990.

[14] J. A. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[15] L. Weng, F. Menczer, and Y.-Y. Ahn. Predicting successful memes using network and community structure. *arXiv preprint arXiv:1403.6199*, 2014.