Predict Startup Success using Network Analysis and Machine Learning Techniques

Bo Guang Huang *

*Stanford University

Abbreviations: ML, machine learning; VC, venture capital, EV Eigenvalue, PR PageRank

Introduction

There is a general believe among the tech communities that the probability of joining a tech startup that will ultimately succeed is about 1 in 10. Whether this statement is true or not, it conveys the notion that failure rate among tech startup companies is very high. The prior work I reviewed mainly focused on link prediction, which translates to if a VC will in the future invest in a Startup. The goal of this research is to use various network analysis techniques on the investment networks, and develop an algorithmic model in which we can predict a given startup will be successful. Success in this research will be strictly defined as startup that is Acquired, IPO, or valued at $1B or more (Unicorn). The success prediction model will make use of various attributes below that demonstrate their effectiveness at ascribing to startup successes

1. Status assessment for for Startup companies base on Centrality calculations (Degree, Betweenness, Closeness, Eigenvalue, PageRank)
2. Investor Status attributes that may contribute to the success of a startup
3. Features that has Power Law distribution, such as amount of funding and location, that could be indicative in finding successes

Literature Review

Networks of venture capital firms in Silicon Valley [1]. Castilla in his study offers an explanation on why venture capital community in Silicon Valley California enjoyed higher amount successes, in term of financial gains from startup investments, when it is compared to a VC community of similar size, specifically Route 128 in Massachusetts. The authors illustrates this regional advantages by demonstrating 2 key differences in the structure of social networks base on the cooperation within the venture capital communities:

1. Collaboration in the Silicon Valley venture capital firms are more dense and also included more connected cliques [1] than that of Route 128
2. The investments made by Silicon Valley venture firms were targeted more toward local startups versus startups that were outside of the local region.

To illustrate the first point, the author applied network analysis methods of calculating centrality measures and finding maximal complete sub-graphs on the two network created based on investment collaboration. The networks, $G =< V, E >$, for each region are constructed where V represents the VC firms, and E represents the collaboration between the firms; this is strictly defined as common investments made together on 1 or more startup company. The author then calculated centrality values for Degree, Closeness, and Betweenness for each V in the networks and compare the statistics in regards to Mean, Standard Deviation, Min and Max. The comparison results show that Silicon Valley VC community clearly has higher values for all 3 centralization measures and this can be interrupted as evidence of overall higher amount of collaboration at the individual VC firm level.
However, the author further believes there are higher amount of collaboration not only on an individual firm level but also within group of VCs. To illustrate this, the paper utilized the method of finding maximal complete sub-graph of three or more nodes. A maximal complete sub-graph is defined where any 2 members within it are adjacent to each other and there are no other nodes that are also adjacent to all the members of the sub-graph. The author defines this as cliques in the VC social network. In this context, cliques are a group of VC firms all of which cooperate with each other in investing common startup companies while no other firms cooperated with all the members of the group. In addition, the author makes use of the amount of n-clique, where n is the maximum path length connecting all the firms in the clique. For example, a 2-clique is one where all members are directly adjacent or linked indirectly through a common in-between firm. The comparison results between the two regional VC communities were even more striking even when the number of VC firms is similar. In Silicon Valley the number of cliques of minimum size 3 is 45 compare to just 12 in Route 128, and 2-cliques of minimum size 10 is 70 compared to just 1 in Route 128. Again, this demonstrate the author's hypothesis of a much more amount of collaboration within the Silicon Valley VC community when compared to Route 128.

Power-law distributions in empirical data [2]. Clauset et al. present in this paper a principled statistical framework [2] for recognizing the existence of power-law within a given set of empirical data. Power law is defined mathematically when a quantity x is drawn from a probability distribution $p(x) \propto x^{-\alpha}$ [2] where α is a constant defined to be the scaling parameter of the distribution. The value for it typically lies within the range $2 < \alpha < 3$. The authors provided a straightforward recipe to discern the existence of power- law within an empirical dataset

1. For step 1, the author suggest plotting the empirical data in a histogram fashion with logarithm scaling on both of the axes, and see if a straight line can be approximated to the distribution. A common mistake to estimate the scaling factor ? is using least square linear regression, since the variance usually in the tail portion of power-law distribution can throw the method off quite a bit. Instead the authors feel that the method of choice for fitting models such as power-law distributions on empirical data is the method of maximum likelihood.
2. For step 2 on testing the power-law hypothesis, Goodness-of-fit test is used qualify the plausibility. The test generates a p-value that measure the distance between the empirical data distribution and the model hypothesized. After fitting the empirical data to the power-law model using Step 1, the author calculates Kolmogorov-Smirnov (KS) [2] statistic for it?s fitting. Next, we will need to generate a set of synthetic

data on the power-law distribution with estimated scaling factor ? and lower bound x_min. We then calculated KS statistic for the synthetic data. The p-value is calculated as the fraction of time the resulting statistic on the synthetic dataset is larger than the value of the empirical data. In this study if $p \leq 0.1$, then the power law hypothesis is ruled out.

3. For step 3 on finding alternative distributions, the author suggest using goodness-of-fit test again on competing distribution, such as power law with cutoff, exponential, stretched exponential, log-normal, and see if we can find a model with higher likelihood comparing to power-law.

In the last section, the author applied the method to 24 different empirical dataset, and 17 to support the fit of power law.

Predicting Investments in Startups using Network Features and Supervised Random Walks [3]. The goal for this study is to develop a system that predicts investment that will be made by a specific investor to a specific startup. This research makes use of Crunchbase investment dataset and maps it as a bipartite graph. The link prediction model is created base on history of prior investment decisions made a long with others as well as existing companies that are already part of the portfolio. The study is broken down into mainly three parts, understanding the statistic of the dataset, developing the algorithm and method for the link predicts, and analyzing the effectiveness of the results. In the development of algorithms and methods, the researchers used mainly two methods, Supervised Learning Algorithms and Supervised Random Walks, both of which uses multiple features from the nodes and the networks. One interesting preferential feature found is that investors with high degree (i.e. making investments) tends to link with startups with low degree, and the opposite for investors with low degree tends to link with startups with high degree. This seems to indicate that investors that are less active tends to more conservative and invest in only companies that are more likely to develop signs of success. The supervised learning techniques used are mostly machine learning algorithms, Support Vector Machine, Logistic Regression, and Bernoulli Nave Bayes. The supervised random walk also makes use of all the various featured discovered,

Link Prediction in Bipartite Venture Capital Investment Networks [4]. Like the previous study in the area of startup investment, the goal of the research is also attempting to use link prediction and make future prediction on specific investor making investments to specific startups. The paper is broken into mainly on Modeling, developing Prediction Methods, and Evaluating Results. Once the graph is established, the team displayed network statistics such as degree centrality for investors and companies, as well as log-log plot of amount in-degree and out-degree for the startup nodes and investors, respectively. There are couple of insight found doing this basic analysis, such as that there is a power-law in the in-degree for the startup nodes which indicates that most companies receives very little number of funding, while there are few who receives quite a lot. Relating to my research, this maybe an indicator of success vs. failure. The startup nodes rank by degree centrality also shown correlation in discovering successful startup, companies such as uber, didi-dache, facebook, airbnb, which all have very large valuation today all rank at the top of weighted degree centrality.

Network Analysis of the Stock Market [5]. The focus of this paper is predicting how stock performance of a public traded company is impacted base on relationships with other companies. The researchers make use of community detection techniques, as well as visualize the graph in open sourced tool, Gephi, as they indicate that visualization is a very intuitive way of finding correlation in graph structures and key market segments. The testing method here is also very interesting, in which the authors look at the overall market performance during the period of US credit crisis between the period of July /2007 and Feb/2009, and how the negative stock performance start at one particular segment of stocks, but quickly permeated throughout rest of market. The first focus of the study is network visualization. It is obvious that each stock can be represented as a node in the graph, however, providing a definition for edge creation is rather not intuitive. The method used here is calculating cross-correlation of stock price changes within a specific time period between 2 different stocks. Link is established base on correlation crossing a certain threshold. Once the graph is established, color is applied to the network visualization where nodes are colored base on Standard Industrial Classification. The team made use of community detection library provided by Gaphi, Fruchterman Reingold algorithm, and the resulting visualization aligned well with the industry classification and coloring. Knowing this is important as it demonstrates that the team is correct with the notion that stock performance correlation is illustrated by identifying market industries. Once the graph is properly constructed, they were able to illustrate at the how the 2007 US credit crisis cascaded throughout this stock market graph.

Methodologies

We will treat this experinment as a supervised learning binary classification problem by giving response labels to startup companies that are successful by status of Acquired, IPO, Unicorn, and response label of 0 to startups with status of Operating and Closed. In order to execute machine learning algorithm with different parameter sets and assess the results in a uniformed fashion, I developed an execution framework in R that does the following:

1. Breaking up the dataset randomly into 90% for training, and 10% for testing the prediction
2. Allowing the creation of different parameter lists that each ML method requires
3. Execution of the ML methods with all different combinations by performing inner for-loops on on values with in each parameters list. Execute each permutation for 3 runs, and take the averages
4. Creating a uniformed test result class that tracks the performances for each glass, which we can further perform analysis and comparisons.

3 categories of supervised learning methods were selected for the experiment and their effectiveness were compared, this includes:

1. Kernel SVM
2. Adaptive Boosting
3. Random Forest

For each of the model created, the following matrices is calculated to assess and compare the effectiveness between each of the experiments.

1. Accuracy: The proportion of true results, both true positives and true negatives among the total number of example

predicted.

$$Accuracy = \frac{TP + TN}{P + N}$$

2. Precision: This is is the proportions of predicted successes that are truly so vs all predicted successes, either correct or incorrect

$$Precision = \frac{TP}{TP + FP}$$

3. Recall: This is the true positive rate measuring the proportion of successful startup that are correctly identified as such.

$$Recall = \frac{TP}{P}$$

4. F1 Score: This measures the prediction accuracy considering both the Precision and Recall, where scores reaching 1 is the best, and 0 the worst.

$$F1 = \frac{2TP}{2TP + FP + FN}$$

5. Matthews Correlation Coefficient (MCC): This takes into account both correct success predictions, incorrect success predictions, as well as successes that were not correctly predicted. This can be seen as a correlation coefficient between the observed results and predicted binary classifications. This returns a value between -1 and +1, where +1 is prefect, 0 is no better than random, and -1 means total disagreement between prediction and observation.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

6. Weighted Cost/Benefits: The problem we are solving here has an uneven Benefit vs Cost ratio, and the purpose measurement is to assess such. The intuition is that investing in a successful startup will provide much greater return vs the associated risk of losing the investment. We will create a score that weight the proportion of true positive prediction higher, with 5x, and subtract the proportion of false positive predictions.

$$WCB = 5 \cdot \frac{TP}{TP + FP} - \frac{FP}{TN + FP}$$

Data Collection and Analysis

There are 3 datasets used in this study sourced from Crunchbase.com, and Techcrunch Unicorn list. They are VC investment data as of 2013, data regarding individual startup company status as of 2015, and current companies that are deemed Unicorns as of 2016. The main methodology in creating the training data for various ML methods is to used investment data from an earlier time period, in this case 2013, and assess the startup success base on information from a later date, in this case status, either acquired or went IPO, as of 2015, or being deemed as Unicorn as of 2016.
The 2013 investment data is used in creating 3 investment network graphs:

1. Investors-to-Startups: This is a bi-partite network where nodes are investors on one side and startups on the other. The edges are represented by investments made from investors to startups.
2. Investors-to-Investors: This is a normalized network created from above with only investor nodes. Edges represent common investments that investors collaborated on.
3. Startups-to-Startups: This is a normalized network created from the 1st with only startup nodes. Edges represent investors that startup have in common.

The following histogram with log-log is also created base on the investment network above. Note that the charts does exhibit a level of Power-Law distribution although I did not go into rigorous assessment of it.

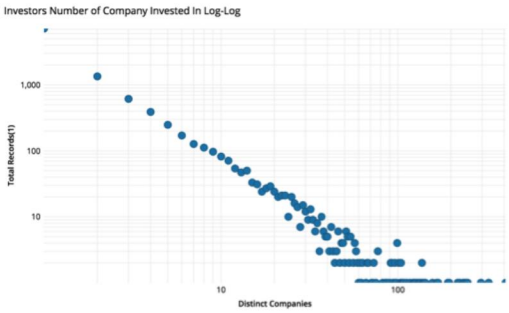

Fig. 1. Number of Distinct Startups Invested in

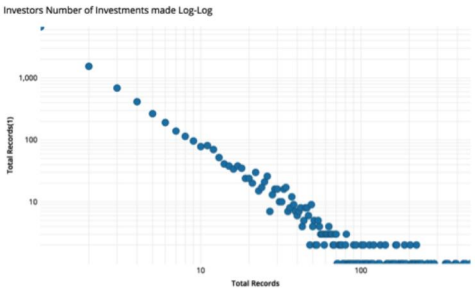

Fig. 2. Number of Investments made

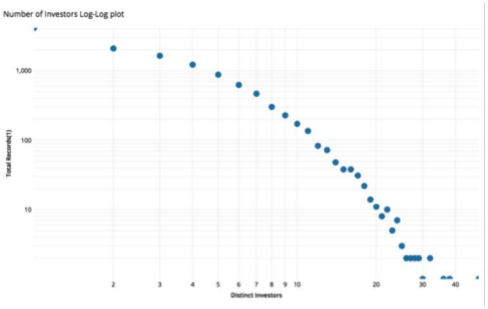

Fig. 3. Number of Investors that Startup has

The next step is to mine for features regarding each startup company that maybe relevant in determining it's future success. As discussed prior, the intuition is that startup with higher status or important within the investment network will tend to be more successful in the future. The intuition that a startup company that holds a high status or importance, i.e. founded by more investors, more prominent investors or

provided with more funding, is indiction that this startup is building a business that the investors trusts in being successful in the future. I will also use knowledge gained from prior studies that location, such as being in Silicon Valley, may attribute in higher success rate. The features selected falls under the following categories:

1. Importance of startups based Centrality scores
2. Importance of investors based Centrality scores
3. Other features that may contribute to startup success: Location, Total Amount of Funding,

Feature Selection

Startup Centrality Scores. The first set of attributes are the various centrality scores calculated for each startup calculated from the Startups-to-Startups network. Startup companies with a higher score here will indict that it is funded by investors who is very active in the investment community, such as invested in a large amount of other startups and/or having a long of collaborating with other active investors. with The following table illustrates the average startups centrality scores calculated for based on the 2013 investment network respective to their most recent status:

Table 1. Startup-to-Startup Centrality Score

Status	Betw	Closeness	Deg	EV	PR
Acquired	11,128	0.3776	0.0188	0.0058	0.000106
IPO	17,947	0.3724	0.0198	0.0045	0.000106
Unicorn	27,814	0.4196	0.0444	0.0166	0.000204
Operating	8,032	0.3457	0.0131	0.0039	0.000085
Closed	5,747	0.3472	0.0123	0.0035	0.000079

Note that the first 3 rows, Acquired, IPO and Unicorn are what is deemed as successful in the model definition. As we can see, there are several centrality scores that are noticeability higher in those categories when comparing to categories not deemed successful, specifically betweenness, degree and pagerank.

This table below also listed top 20 companies base on their degree centrality. Bolded companies are those that are deemed successful startup in that they are either IPO, acquired, or valued at $1B or more.

Table 1.2 Top 20 Startup companies base on Degree Centrality

Company Name	Degree	Closeness	Betweeness	Eigenvalue
Ark	0.138166895	0.474523669	211649.5612	0.056979451
Groupon	0.13123575	0.475434213	185647.4307	0.053461835
Dropbox	0.128134975	0.465107002	120832.6945	0.05530611
Path	0.125490196	0.470937415	172932.7847	0.052460877
Hangtime	0.124304606	0.472562738	150018.306	0.049082795
Swiftype	0.122754218	0.466190487	80469.67072	0.057246072
The Climate Corporation	0.119471044	0.469386741	97324.08455	0.052391946
Twilio	0.118194254	0.467930395	131901.2503	0.048241406
Box	0.117920657	0.472605661	178107.079	0.030358579
Airbnb	0.114911081	0.455697323	53516.48182	0.054724178
Optimizely	0.114546284	0.464463471	152191.2329	0.042847301
CrowdMed	0.114363885	0.459825198	77216.68862	0.055438661
Yammer	0.113178295	0.463615079	145960.4291	0.0457911
YuMe	0.112904697	0.463264215	85994.369	0.046364036
RethinkDB	0.111627907	0.461047541	99470.57075	0.050987457
42Floors	0.109712722	0.46129279	81175.18306	0.053272537
Backplane	0.109621523	0.460578209	150657.0283	0.047180971
IFTTT	0.109347925	0.461948065	70102.50545	0.051398416
Twitter	0.107250342	0.463553123	102464.1753	0.046781961
Stripe	0.107250342	0.452290684	46135.21465	0.052133197

Fig. 4. Number of Investors that Startup has

Investor Centrality Scores. In similar fashion, we can also assess the importance of investor within the network using centrality scores. However, since the goal is to see if the importance contributes to the success of the their invested startup, we go through each startup and create an aggregated centrality score base on the contributing investors. The following table illustrates the average investor centrality scores aggregated on their respective startups:

Table 2. Investor-to-Startup Aggregate Centrality Score

Status	Betw	Closeness	Deg	EV	PR
Acquired	860,516	1.4395	0.0791	0.1416	0.0028
IPO	852,529	1.7419	0.0826	0.1398	0.0030
Unicorn	2,174,025	0.4196	2.6275	0.3704	0.3704
Operating	610,306	1.1215	0.0545	0.0953	0.0068
Closed	559,071	1.0063	0.0494	0.0855	0.0018

Again we see the for startup that are success tend to have higher aggregated investor centrality scores, specifically Betweenness, Eigenvalue.

Funding and Location. The other 2 attributes that will be selected will be Total Funding and Location of the startup. The reason for picking Total Funding is base on finding that there are higher concentration of successful startups that recieves more funding. The reason for picking Location is based on founding in [1] that the region that startup is funded in does have an contributing effect to it's probability of success.

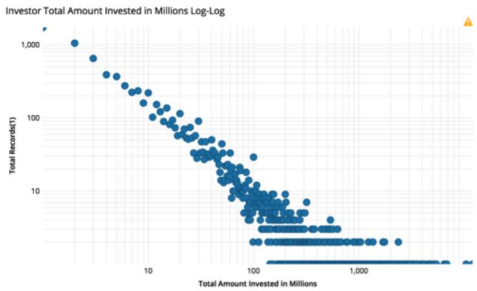

Fig. 5. Investor Total Amount Invested

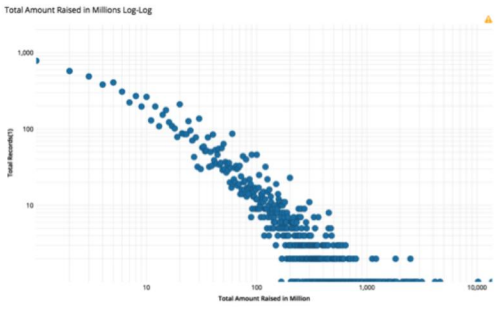

Fig. 6. Total Amount Raised By Startups in Millions

Experiments and Results

Kernel based Support vector machine (SVM). is a supervised ML method thats represents the training data in a n-dimensional finite space, the data point is viewed as a n-dimensional vector, and the model tries to find a separation for the data points with a $(n-1)$-dimensional hyperplane. The separation which picked by the model is base on representing the largest margin between the two binary classes. Once the model is created, subsequent test data are then mapped into the same space and predicted to belonging to a category based on which side of the margin they falls under. The problem is stated in a finite dimensional space, however it is often that the data sets are not linearly separable in this space. Because of this reason, a new method is proposed that the original finite-dimensional space be mapped into a space with a much higher dimension, therefore making the separation easier. By defining the variables in terms of a kernel function $k(x, y)$, the hyperplanes in the higher-dimensional space can be defined as the set of points whose dot product with a vector in that space is constant and in turn ensure computational load is reasonable. An extension of the kernel based SVM, is called Support vector clustering, or SVC, that first look in the feature space for the smallest sphere that encloses the image of the data points. It is then mapped into the data space using the kernel function, which forms a set of contours which encloses the data points. The contours therefore are defined as cluster boundaries.

I have experimented with using different SVM method using types of C-SVC and nu-SVC and kernel types, rbf and lapcedot. The following tables shows the following in which model highest score is picked to represent the class of test models:

Table 3. Kernel based SVM results

SVM	nu-SVC	nu-SVC	C-SVC	C-SVC
Kernel	rbf	laplacedot	rbf	laplacedot
# of Models	24	24	60	60
Accuracy	0.43	0.68	0.68	0.70
Precision	0.27	0.43	0.58	0.56
Recall	0.60	0.34	0.07	0.12
F1	0.38	0.38	0.13	0.19
MCC	-0.03	0.18	0.11	0.15
WCB	2.39	1.54	0.55	0.55

In the result above, it looks to be that even though nu-SVC with rbf kernel has the highest weighted cost benefit, WCB, score, it is nu-SVC with laplacedot kernel that has the highest overall performance results.

Adaptive Boosting. is a supervised ML meta-algorithm that uses the outputs of a number of learning algorithms, termed weak learners, and combined them into a weighted sum that represents the final output as a boosted classifier. The adaptive nature of AdaBoost is that weak learners are weighted to supplement cases that are misclassified by other classifiers, therefore this algorithm is sensitive to noisy data and outliers, and can be less susceptible to the overfitting issue experienced by other learning algorithms. Even though the individual learners can be weak, as long as the performance of each learner is better than random guessing, the final model model will converge to a strong learner. AdaBoost can described as

$$F_T(x) = \sum_{t=1}^{T} f_t(x)$$

where each f_t is a weak learner that takes a data point x as input and outputs the predicted class. For each data point in the training set, each weak learner produces an prediction $p(x_i)$. At each iteration t, a weak learner is selected based on the minimum training error sum E_t of the resulting t-stage boost classifier and is assigned a coefficient α_t such that

$$E_t = \sum_i E[F_{t-1}(x_i) + \alpha_t p(x_i)]$$

Where F_{t-1} is the boosted classifier that was built up during previous stage of training. $E(F)$ is the error function and $f_t(x) = \alpha_t p(x)$ is the weak learner that is being considered for addition to the final classifier. At each iteration of the training process, a weight w_t is assigned to each sample in the training set equal to the current error $E(F_{t-1}(x_i))$ on that sample. These weights is used as how much each weak learner contributes to the final classifier.

The AdaBoost prediction also provides the probability score for classification, and I was able to adjust it to see the outcome in which to boost the overall Weighted Cost Benefit score.

Table 4. AdaBoost Result

Prob. Threshold	0.5	0.4	0.3
Accuracy	0.70	0.67	0.57
Precision	0.51	0.44	0.33
Recall	0.12	0.45	0.55
F1	0.20	0.18	0.15
MCC	0.13	0.11	0.10
WCB	0.58	1.96	3.02

As the result shows, even though the overall accuracy goes down while threshold is lowered, the WCB score does increase in this case. This indicts that the classification problem is an uneven one, and I'll need to consider more into giving more weight to success prediction class given that the benefit is much higher here.

The AdaBoost method also provides an interesting statistics of the importance of the features attributing to predicting the output, and the following shows the top 5 feature with ranked relative importance.

Table 5. Feature Importance

Features	Importance
Region	50.2687078
Startup To Startup Closeness Centrality	15.3128559
Total Funding	11.9721351
Investor To Startup Aggregate Centrality	12.5053595
Startup To Startup Betweenness Centrality	3.7827085

Random Forest. is a ensemble learning method which during training time constructing a multitude of decision trees. When subsequent test data is provided, the output classification is the class that appears the most often among the classes provided of the individual trees. This model usually correct for the decision trees' habit of overfitting to the training set. The training algorithm applies the general technique of boostrap aggregating, or bagging, to tree learners. Given a training set $X = x_1, ..., x_n$ with labels $Y = y_1, ..., y_n$, the algorithm performs bagging repeatedly, for B times, selects a random sample with replacement of the training set and fits trees to these samples:

1. For $b = 1, ..., B$:
2. Draw a boostrap sample Z of size N from the training data
3. Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

   (a) Select m variables at random from the p variables

   (b) Pick the best variable/split-point among the m

   (c) Split the node into two daughter nodes

4. Output the ensemble of trees $[T_b]_1^B$

To make a prediction at a new point x: Let $C_b(x)$ be the class prediction of the bth random-forest free. Then $C_{rf}^B(x) = majorityvote[C_b(x)]_1^B$

I have applied Random Forest method with various parameters, such as Number of Trees and Node Size. A total of 51 experiment models was created, and the following table list out the top 4 results with the used parameters.

Table 6. Random Forrest results

# of Tree	600	600	1000	800
Node Size	3	4	2	3
Accuracy	0.72	0.72	0.71	0.69
Precision	0.51	0.50	0.48	0.52
Recall	0.22	0.20	0.22	0.21
F1	0.31	0.29	0.31	0.30
MCC	0.18	0.18	0.17	0.17
WCB	1.03	0.96	1.02	0.96

From the result it looks Random Forrest generally performs well with accuracy around %70 and MCC around constantly around 0.18. However, the WCB score is generally law, meaning that the chance of predicting successful startup is relatively low.

Conclusion

My initial pre-conception of tackling this problem and expecting to some how achieve high success prediction accuracy still remains elusive. Even when having accesses to an array of powerful machine learning tools, making correct prediction on real-world applications can still be quite challenging and difficult. However, through much of the struggle, I feel I managed to learn quite a bit on the various machine learning techniques, and much of the building blocks and technology in tackling difficult prediction problems. In regards to tackling this specific problem, I feel that the testing framework constructed here is a foundation that can be extensible upon for future improvements. By incorporating more sources of information, larger pool of training data, and gaining more experiences in machine learning, I feel this is a problem that can be tackled and eventually be build into production use.

1. Networks of venture capital firms in Silicon Valley: Emilio J. Castilla. 2003
2. Power-law distributions in empirical data: Aaron Clauset, Cosma Rohilla Shalizi, M. E. J. Newman. Jun 2007 (v1), last revised 2 Feb 2009 (this version, v2)
3. Predicting Investments in Startups using Network Features and Supervised Random Walks: Arushi Raghuvanshi Tara Balakrishnan Maya Balakrishnan. 2015
4. Link Prediction in Bipartite Venture Capital Investment Networks: Charles Zhang, Ethan Chan, Adam Abdulhamid. 2015
5. Network Analysis of the Stock Market: Wenyue Sun, Chuan Tian, Guang Yang. 2015