Predicting the Spread of the CS Major at Stanford

Udai Baisiwala
Economics
Stanford University
udai@stanford.edu

Frederick Robson
Symbolic Systems
Stanford University
frobson@stanford.edu

Kevin Rakestraw
Computer Science
Stanford University
krake@stanford.edu

December 11, 2017

Abstract

In this paper, we explore the degree to which student majors can be predicted by the majors of their connections. Specifically, we consider a unique dataset of every Stanford Alumni and use the groups of people with whom each alum shared activities and residences as an approximation of their friends. Then, we attempt to predict whether students will be CS majors based on the major distribution of their friends. We attempt to predict this using naive Bayes, linear regression, and SVMs and also model choice of major using an SIR disease-spreading model and compare the results of these four methods.

Introduction

As undergraduates at Stanford, we participate in a variety of activities and live on campus for most of our time in university. We personally found that the friends we made through student activities and dorms greatly influenced the choices we made with regards to major and careers. Given this experience, we were interested to see how effectively we could predict someone's major by looking only at the majors of people with whom they shared activities and residences.

Fortunately, Stanford has an alumni database that includes information on residences and activities for a significant proportion of graduates. We were able to scrape that database and generate a dataset of alumni with information on year, gender, current city, major, degree, activities, and residences at Stanford. We then use several machine learning methods and an SIR disease model to attempt to predict whether each alum was or was not a CS major.

Related Work

Kermack & McKendrick (1927) originally proposed the SIR Model to explain the spread of epidemics, thereby creating a powerful tool for modeling disease. Since their original paper, the model has been used to effectively build insights into disease and predict the spread of a variety of diseases. For example, *Huang et al (1990)* used a a multi-group SIR model to effectively model the spread of HIV at the height of the Aids epidemic and*Shulgin et al (1998)* used the SIR model in order to evaluate the effectiveness of 'pulse' vaccinations for measles. Using the SIR model they were able to prove that with relatively low numbers of vaccinations within an at risk community, a disease such as measles could be essentially eradicated. We were interested in whether the SIR model could extend beyond diseases, and to modeling the spread of Computer Science Majors.

The SIR Model has previously been used to model much more than the spread of an epidemic. While the SIR Model works particularly well for epidemics in how it models the intra-personal stages of infection as well as the interpersonal spread on the disease, it can be applied to any problem that can be reflected by intra-personal stages and interpersonal spread. In particular, the SIR Model has been used to model the spread of information, rather than a virus. We found that these papers proved more relevant to our attempts to model the spread of the Computer Science major.

Bettencourt et al (2005) attempted to model the spread of the Feynman diagrams through the theoretical physics communities in the USA, Japan and the USSR from their invention in 1949 to 1954. They identified someone as "infected" in a particular year if they cited papers by Feynman or used some of Feyman's techniques. To create edges they recreated a network of contacts - author by author - using unpublished correspondance, preprints and lecture notes alongsidemore recent interviews and published recollections. They applied three different models to their dataset: the SIR Model, the SIZ model, and the SEIZ model. THe SIZ model creates an alternative infection to compete with the infection in the SIR model, and the SEIZ model additionally creates another category: incubators. Incubators have been "infected," but for a certain amount of time, they are not yet able to infect others.For each of these models, they estimated the parameters that minimized the average absolute value of deviation between number of infected individuals in a time period and the mean number of infected indiviudals from their model.

Their models produces a number of interesting results. For example, the model is able to capture the fact that use of Feymann Diagrams only took off in the USSR after the "Atoms for Peace" initiative, where for the first time scientists from the USA and USSR met face to fact. The model is also able to capture that the US has a greater increase in susceptible populations as a result of rapidly increading enrollment in Physics PhD programs. More broadly, they find that the spread of Feymann diagrams appears analagous to a very slowly spreading disease, with characteristic progression times of years instead of days or weeks. The model also finds that there are consistently long recovery times. They attribute these long recovery times to the fact that it is ver difficult to "recover" from an idea.

Meanwhile, *Zhu and Ying (2014)* recently applied the SIR model in order to detect the initial source of an information spread across a network. They were able to build a network topology from a snapshot of the network, and identify infected nodes. However they have no way from the snapshot of knowing the difference between susceptible and recovered without previous knowledge. However using this snapshot they were able to create a reverse-infection algorithm based on the SIR model in order to approximate the location of the source of the information.

Their reverse infection algorithm ultimately led to substantial results. They were able to stay find an estimated source within a constant distance from the actual source with a high probability. They showed not just how SIR model can be very relevant outside epidemic outbreaks, but that with limited knowledge of an network the SIR model can be applied to understand how a network came to be its current state.

Data Collection

We collected our data from the Stanford Alumni Database. The Stanford Alumni Database has collected information on graduates of Stanford University since Stanford's inception and is available to all current Stanford students and all Stanford Alumni. The database contains 274,320 profiles of Stanford graduates from 1890 to the present day, covering undergraduates, MBA's, PhDs, master's students and more.

Each profile contains a varying amount of information. Older profiles have very little information and are mostly stripped down to the bare essentials, whereas more recent profiles contain much more information. For example, the profiles from the 19th century only include name, year, major and Greek organization (if applicable). The more recent profiles may also include residence on campus for each year, student activities for each year, current address and job history, though not all alumni profiles contain all this information.

We developed and utilized a web scraper in order to extract the alumni information from the Stanford Alumni Database. With multiple threads, we were able to scrape all of the data in roughly 50 hours.

As we sought to model the spread of computer science along the student population using our scraped data, we ultimately decided to filter out much of the database to focus on the more recent years where much of the data is populated. Thus, we filtered out all students that graduated before 1980.

Graph Construction

For our problem we sought to model the spread of the computer science major among the student population from 1980 until now. We identified all our nodes as students from 1980 until now, and decided we would try two different factors to represent our edges: housing and activities.

Within the scraped data, housing and activities were the two most tangible indicators of a connection between people. So we decided to build connections by seeing who shared housing and activity information during their time at Stanford. In doing so we had to take a slightly different approach for both.

The majority of housing data instances had an associated start and end date that reflected the time a person was in that specific housing. We were able to detect instances of users living together by seeing if there was an overlap in when they stayed at a specific housing assignment. Using our housing data we were able to find all the housing connections between different alumni.

Some activity data instances included start and end information as included with housing, but these were very few. For the most part activities were not given start and end information so we had to model activity connections in a more naive way. Instead of specifically seeking overlap, we dictated that connections would be made if two people were in the same activity and they shared time together at Stanford.

We have four decades of computer science majors in our dataset. However, when modeling this using the SIR model, students are only susceptible to infection or capable of infecting others if they are currently at Stanford. Therefore, we filtered the graphs for each year so that, for instance, the only nodes active in the graph in 2000 were students currently at Stanford in 2000.

One other key observation about our model, is that the graph is directed. As most students are undeclared until sometime between their sophomore year and the start of their junior year, there wouldn't be a reason why a freshman would have an influence on a senior for example. Thus, we chose to model our graph as a directed graph where edges go from older students to younger students.

We outline the activity and housing degree distributions of the graphs in Figure 1 below. Both graphs have 59,718 nodes - the number of undergraduate students between 1980 and 2015. The housing graph has a total of 7,181,993 edges and the activity graph has a total of 19,093,459 edges.

The activity graph is of higher average degree for a few reasons. First, as mentioned above, we didn't have specific date information for activities, so we had to assume students participated for all four years. This essentially means that for each activity one participates in, one has edges going to everyone who participated in the activity the same year and the next three years, while for housing one only has edges going to the people who lived in the same house at the same time. Second, when we looked at the underlying data, we found that average residence size was smaller than average activity size and had a lesser standard deviation. This again helps explain the lower average degree and also helps explain why the activity out-degree distribution is more even than the housing out-degree distribution, where most nodes have a degree less than 500.

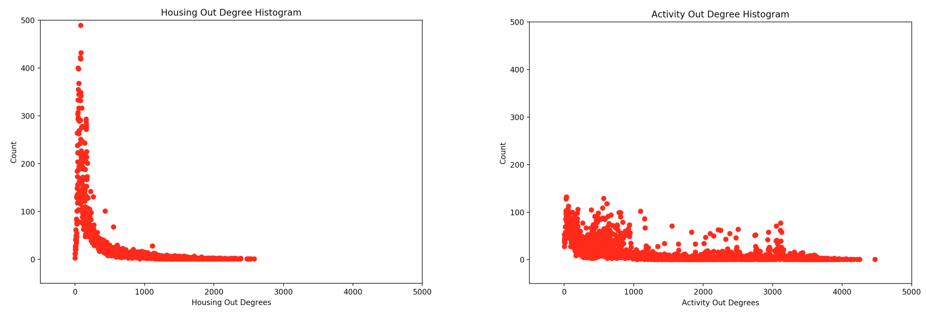

Figure 1: Degree Distributions of the Housing and Activity Graphs.

SIR Model

The SIR model (Kermack & McKendrick, 1927) models the spread of disease through a community. Essentially, the model defines a probability of infection, representing the probability that an infected person will infect someone they are connected to, and a probability of recovering, representing the probability an infected person recovers. Then, using different community structures and estimates of virulence, one can estimate the spread and impact of a disease on a population.

Given the explosion of interest in computer science at Stanford over the past 20 years, we were interested in exploring the relevance of this disease model to the spread of computer science as a major. Figure 2, below, illustrates this growth over 1990-2015. Here, we observe that growth was especially rapid over 1991-2001 and over 2007-2015. Given that epidemics are the rapid growth of a disease, these are times when the SIR might be especially appropriate.

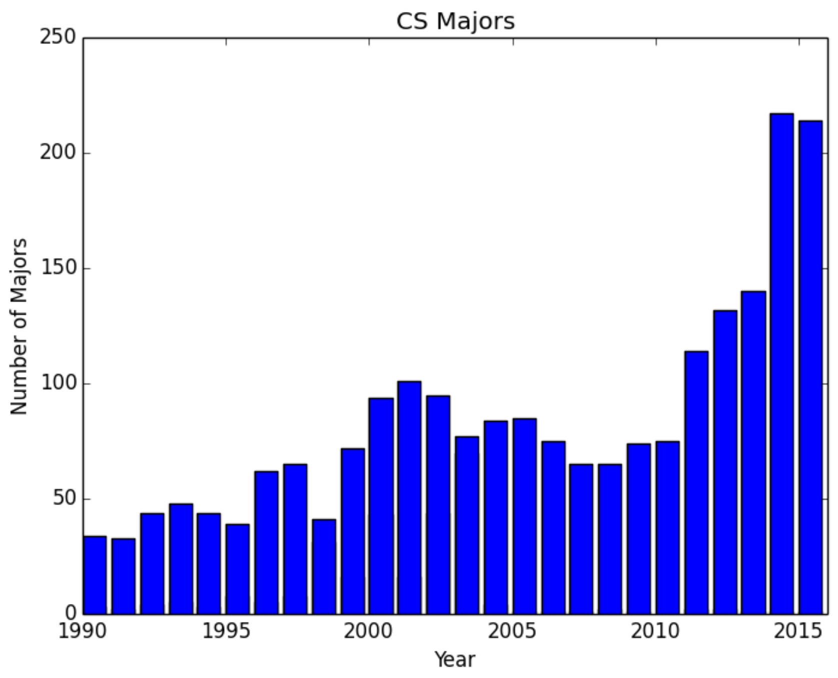

Figure 2: Growth in the CS Major at Stanford

Our SIR model categorizes people as either infected, recovered, or susceptible as follows:

- Initially Infected: The juniors and seniors who were majoring in CS at the starting year of the model.
- Probability of Infection: β
- Recovered: CS majors who have graduated (and can therefore no longer infect people)
- Susceptible: All Stanford undergraduates who are freshmen or sophomores in year i of the model

Because we are modeling Stanford undergraduates, we don't need to include every transition probability included in a typical SIR model. Specifically, since recovering is equivalent to leaving Stanford, we don't need to define a probability of recovering and can instead declare people recovered when they graduate. Therefore, we only have one unknown variable - the probability of infection $= \beta$.

We use the SIR Model to attempt to predict two outcomes. First, we want to attempt accurately predicting whether each individual becomes a CS major or not. Second, we want to see if we can accurately predict the growth of the CS major irregardless of whether we correctly predict the exact people who pursue CS. Thus, we define different loss functions for these two objectives so we can see how our model performs when optimally tuned for each of these aims.

To find the optimal β for predicting growth in the CS major, we define $Loss_{(1)}$ as follows:

$$Loss_{(1)} = \frac{1}{N} \sum_{\text{y=Start Year}}^{\text{End Year}} \frac{|T_y - (1/15)\sum_i^{15} \text{SIR Model}_{y,i}|}{T_y}$$

Where N is the number of years, T_y represents the true number of CS majors in year y and SIR Model$_{y,i}$ represents the predicted number of CS majors in year y in iteration i of the SIR model. The loss function captures the percentage by which we are wrong for number of CS majors each year and then takes the average across all years.

To find the optimal β for predicting whether individuals are CS majors or not, we define $Loss_{(2)}$ as follows:

$$Loss_{(2)} = \frac{1}{N} \sum_{u \in U} \begin{cases} 100(1 - \frac{1}{15}\sum_i^{15} I_{i,u}) & \text{u is a CS Major} \\ \frac{1}{15}\sum_i^{15} I_{i,u} & \text{u is not a CS Major} \end{cases}$$

Where N is now the number of undergraduates, U is the set of all nodes, and $I_{i,u}$ represents whether node u was infected in iteration i. This loss function captures whether we are correctly classifying each undergraduate.

With this loss function, the penalty for misclassifying a CS major is $100\times$ the penalty for misclassifying non-CS majors. This is because only around 2% of the population are actually CS majors, so if we don't increase the weight on misclassifying CS majors, the optimal β is 0 and the model predicts no one is CS. Given that it is difficult to evaluate the meaning of this loss, we also provide a % misclassified in our results, which is equivalent to $Loss_{(2)}$ without the weighting of misclassified CS majors.

Machine Learning Methods

In order to better understand the performance of our SIR model, we also implemented and ran linear regression, a support vector machine, and a Naive Bayes predictor on this dataset. We used the sklearn package for the SVM and Naive Bayes classifiers and used numpy builtins to implement linear regression.

Specifically, for each person we are able to get year of graduation, gender, number of connections in the housing graph, number of connections in the activity graph, proportion of connections in the housing graph that are CS, and proportion of connections in the activities graph that are CS. We can use these features to train classifiers and then evaluate the efficacy of those classifiers using the loss functions defined above and the % misclassified. In order to more directly compare the efficacy of this model against the SIR model, we also train and evaluate classifiers that only use information from the housing graph or only use information from the activity graph.

Results

As discussed previously, we want to evaluate (1) our ability to predict the correct number of CS majors per year and (2) our ability to predict exactly who becomes a CS major.

Predicting Number of CS Majors per Year

We begin by discussing our results for that first question. Given that CS has not been spreading at the same rate consistently for 1990-2015, we decided to evaluate performance both on that overall time period and the few selected sub-periods where the CS major appeared to actually demonstrate rapid, epidemic-type growth. In order to evaluate the appropriateness of the SIR model for modeling this, we earlier defined a loss function, $Loss_{(1)}$, as the average percent by which we are wrong in predicting the number of CS majors each year. In Figure 3, we illustrate the $Loss_{(1)}$ we attained for the housing and activity graphs over various time periods.

We observe that we are better at predicting the number of majors per year for shorter time periods than for longer ones in the housing graph. This makes sense - since we are only feeding

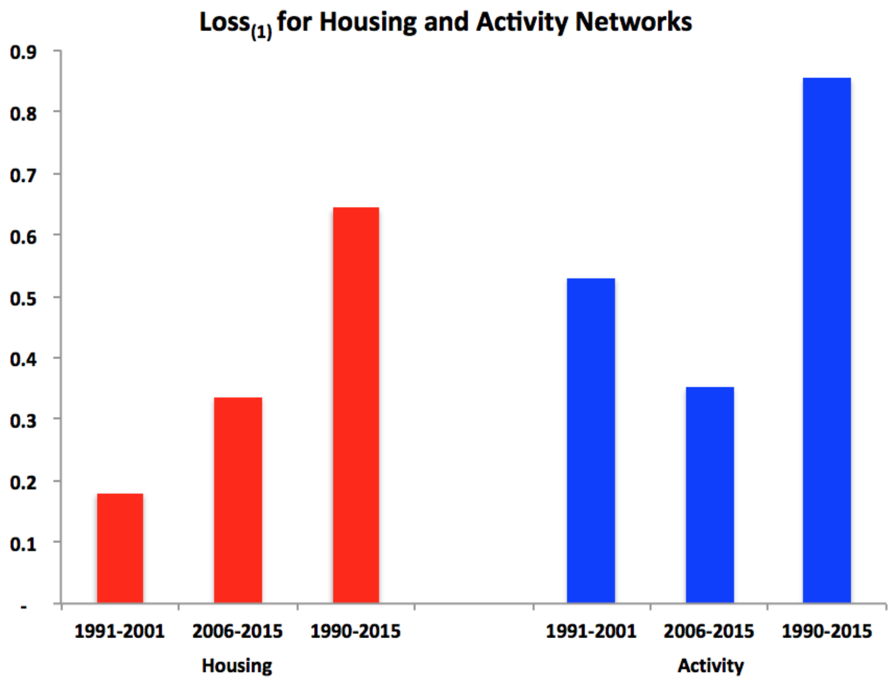

Figure 3: $Loss_1$ for the activity and housing networks with the SIR algorithm run over different time periods

the model the set of CS majors in the first year, there is more of a probability of differing greatly from the actuals over longer time periods. Surprisingly, in the activities graph, we are better at predicting growth over 2006-2015 than over 1991-2001.

Overall, we were able to get losses much lower than we expected. While a loss of .9 might seem high at first glance, this really means that we are wrong by an average of 90% when guessing the number of CS majors. Given that the average proportion of CS majors across the entire dataset was 3.5%, getting the number of CS majors per year correct to within 2x seems reasonable. We were also very pleased to observe the performance of the SIR model on the housing graph over 1991-2001. We saw $Loss_{(1)} = .1804$ on that subset of the data, which is a very accurate prediction of the growth of the CS major. The performance of the SIR model on those years is further illustrated in Figure 4, which shows the predicted and actual number of CS majors per year. We observe that here β, the probability of infecting someone, is .054, which is extremely low compared to typical SIR models. This makes sense given that the average degree of nodes in the housing graph is 120 and that the growth in the number of CS majors is not exponential in real life.

One explanation for the efficacy of the model over 1991-2001 is that in that period, the Computer Science Department was less well known than it is nowadays. In 1991-2001, a freshman or a sophomore might have been convinced to take an introductory CS class because one of their older friends was a CS Major. Today, the majority of freshmen come into Stanford planning on taking CS106A. Therefore, whether or not you have friends who are CS majors is probably less influential today than it was.

We were also interested in understanding why running SIR on the housing network is generally a better predictor than running SIR on the activity network. Empirically, from our experience at Stanford, we have found that our experiences (especially in our freshman and sophomore year when undergraduates are "susceptible") are shaped by our residences more than our activities. It is hard to live in a small dorm and not become friends with a number of dorm-mates. By contrast, there is a much wider range of experience in activities. Some clubs might be very influential, but many clubs have hundreds of members and therefore you might never meet all the members of your club. A CS major in your club who you never meet is unlikely to influence your decision. Additionally, the data for the activities network is less thorough than the data for the Housing Network. The alumni database will often only state that an undergrad participated in an activity, not give the

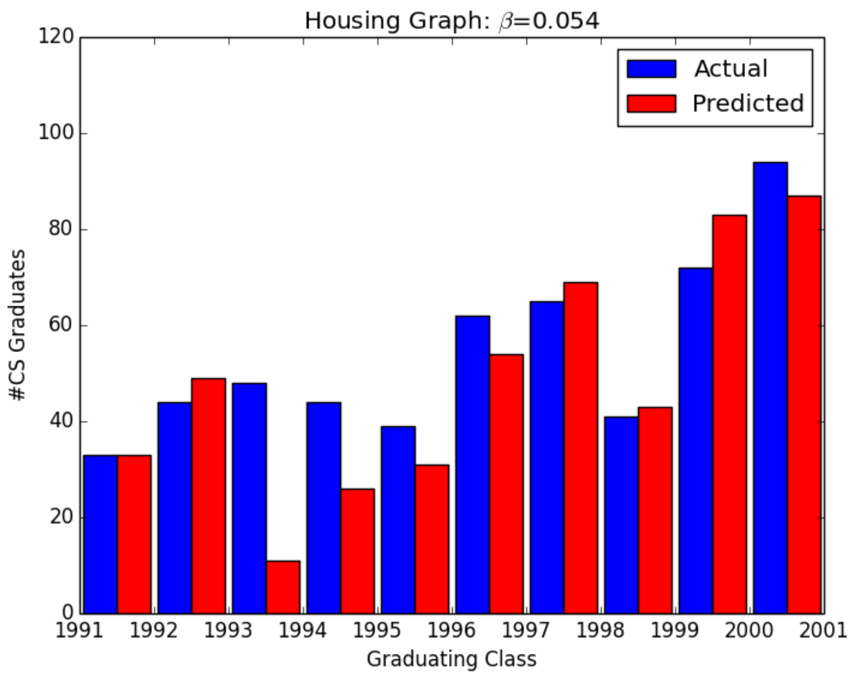

Figure 4: Predicted and actual number of CS majors per year over 1991-2001 using the SIR model

dates. As a result, if there are no dates, we have to assume that the undergrad participated in the activity for all four years. Obviously, this assumption is often wrong. Therefore, there are certainly a number of edges that erroneously exist in the graph.

To summarize, we find that the model is not able to predict the number of CS majors to an extremely granular accuracy, but it does predict a number of majors per year that is within 2x the actual number even when running the model over a 25 year period. This is quite impressive given we are only giving the model the set of people studying CS in the first year of each model.

Predicting the Major of each Individual

Next, we want to attempt to predict the major of each individual. Here, we ran several different machine learning algorithms using the data from our graph. We also again attempted to use the SIR model as a classifier using the weighted loss function discussed in the SIR Model section. In reporting our results, we use the % classified or % misclassified to compare the accuracy of the different classification strategies.

First, we evaluate some initial results from when we optimize the SIR model for $Loss_{(2)}$, in Tables 1 and 2 for the housing and activity networks, respectively. First, we note that for the different graphs, the true % CS majors is 3-7% for different time periods and different graphs. (Since we don't include nodes if we don't have any data on them, the number of nodes and number of CS majors is different across the activity and housing graphs for the same time period.)

Time Period	Beta	Loss	CS Classification	Non CS Classification
1991-2001	.052	1.63	3.28%	99.9%
2006-2015	.052	1.02	45.4%	91.7%
1990-2015	.054	0.92	51.2%	88.9%

Table 1: Percent CS and Non-CS correctly classified in the housing network

Then, let us consider Table 1, the results for the housing network. We were surprised to observe that the classifier is quite bad at identifying the true CS majors over 1991-2001, given that SIR run on the housing network over 1991-2001 was best able to predict the true number of CS majors each year out of all the graphs we attempted. This result might also be a result of our scoring function. Despite the fact that we weigh correctly classifying CS majors at 100x the importance of

Time Period	Beta	Loss	CS Classification	Non CS Classification
1991-2001	.053	1.49	17.2%	93.4%
2006-2015	.053	1.52	13.4%	95.2%
1990-2015	.054	1.26	35.4%	85.2%

Table 2: Percent CS and Non-CS correctly classified in the activity network

correctly classifying non-CS majors, the fact that there are far more non-CS majors in the dataset means that the optimal model has a low beta and predicts very few people are CS majors, which is what happened in this case - 99.9% of non-CS majors were correctly classified, so the model clearly is biased towards identifying most people as non-CS majors.

We observe that over 1990-2015, the model correctly classifies 51.2% of CS majors, in a population where 7.2% of people were CS. The model is overestimating the number of CS majors significantly - it is classifying more than 10% of the non-CS majors as CS. But we still found it extraordinary that a model, given only the set of people studying CS in 1990, could correctly identify more than 50% of CS students over the next 25 years using only data on student cohabitation over that time period. Over 2006-2015, results were quite similar to 1990-2015.

Next, let us consider the activity network. We observe that here, the % of CS majors correctly classified is lower than the corresponding percentage from the housing network over 2006-2015 and over 1990-2015, and the percentage correctly classified is quite low in absolute terms over 1991-2001 as well. This reinforces our perception from experimenting with predicting the number of CS majors per year that connections in the housing network more strongly influence people to pursue a specific major than connections in the activity network.

Comparing SIR Results to Machine Learning Results

We also trained several predictors on the dataset and attempted to use those to predict whether people pursue CS or not. We ran a linear regression, trained an SVM and trained a Naive Bayes classifier. As discussed earlier, for predicting using only the activity graph or only the housing graph, we used features [year, gender, proportion of connections who are CS, number of connections]. When we predicted using both graphs, we used features [year, gender, proportion of connections who are CS in activity graph, number of connections in activity graph, proportion of connections who are CS in housing graph, number of connections in housing graph]. In Figure 5, we compare results from these machine learning techniques and the SIR model. Here, we consider % Misclassified as the relevant metric to compare results.

In running this analysis, we noticed that our Naive Bayes classifier performed badly relative to the other methods on the activities graph and when we gave our classifiers all data. We also observed that while most of the classifiers saw their performance deteriorate when given only the housing network, the Naive Bayes classifier performed best there. We suspected this might be because the proportion of CS majors in connections on the housing graph is actually really significant in determining a person's major. When we looked at the results of our regression on all data, we found a large, statistically significant coefficient on the proportion of CS majors in connections on the housing graph, which supports this hypothesis. If this is the case, it would help explain the Naive Bayes results, since the classifier might mostly be considering the proportion of CS majors in connections. However, this doesn't explain why performance deteriorates significantly for Naive Bayes on all data.

We observed that the SIR model actually performed quite well relative to the other classification models. We initially suspected this might be because the SIR model simply used a small beta and predicted that almost no one was a CS major, but Tables 1 and 2 show that the SIR model is actually correctly classifying both CS and non-CS majors for most of the graphs and therefore is actually classifying to a substantially accurate manner.

Of the machine learning methods that we explored, the SVM consistently performed best. However, the SIR performed similarly well for a surprising proportion of the graphs. Similar to the other experiments, we were surprised to observe how well the SIR performed relative to the machine learning methods. The SVM trains on a dataset with the true proportion of CS neighbors for each node, but the SIR model performs similarly well despite the fact that we train it with far less data. We were pleased with the performance of the model relative to our machine learning comparisons.

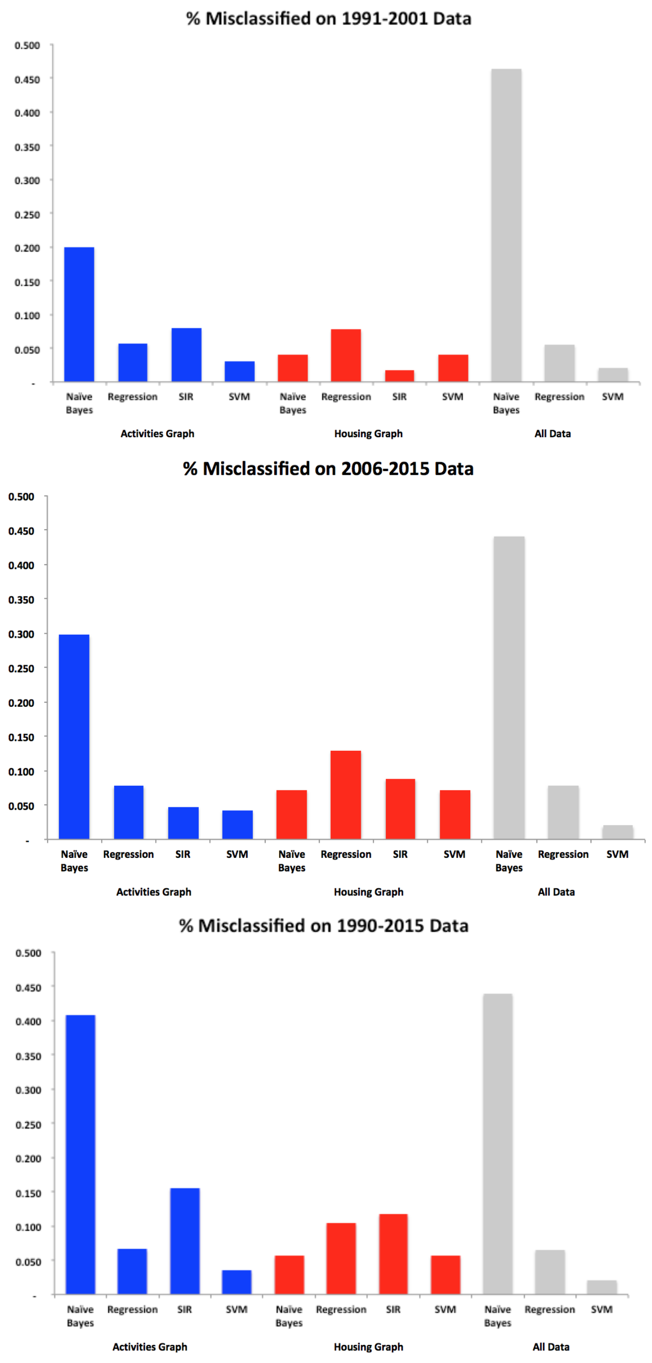

Figure 5: Percent misclassified across various time periods and algorithms

Conclusions

Inspired by how we have been impacted by our peers at Stanford, we used an SIR model to attempt to approximate the spread of the CS major at Stanford. We used an SIR Model to approximate both the number of CS Majors graduating each year, as well as who will go on to become a CS Major. We then compared our results to those from a series of different machine learning approaches.

Even though the best machine learning approaches outperformed our SIR model, we were impressed by how effective our SIR model was. That it was able to effectively capture the spread of computer science across Stanford - especially in the period 1991-2001 on the housing network - suggests that the CS major is somewhat "virus-like." Our study expands on previous work by showing that the SIR model can effectively be used to estimate the spread of information similarly to the spread of a disease.

For future research, it would be interesting to study whether other phenomena from the Stanford community can be modeled as a disease. For example, it would be interesting to see whether certain activities are like viruses and gain huge popularity for a short period of time before disappearing from campus. Similarly, we would like to see whether our findings extend to other universities - could the uptake of computer science at Harvard be modeled in the same way as we have at Stanford?

Team Contributions

Frederick Robson: Coding and running scraper to collect data from alumni.stanford.edu, SIR code and write-up, previous-work write up
Udai Baisiwala: Parser for scraped data, coding/running machine learning techniques and wrote results write-up
Kevin Rakestraw: Construction of graph from alumni data, Poster write-up, previous work write up

Bibliography

Bettencourt, L.M.A., Cintrón-Arias, A., Kaiser, D.I., & Castillo-Chávez, C. (2006). The power of a good idea: Quantitative modeling of the spread of ideas from epidemiological models. Physica A: Statistical Mechanics and its Applications, 364, 513-536. DOI: 10.1016/j.physa.2005.08.083

W. Huang, K. Cooke, C. Castillo-Chavez,(1990), SIAM J. Appl. Math. 52 835–854.

Kermack, W.O., McKendrick, A.G., (1927). Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences

Shulgin, B., Stone, L. & Agur, Z. Bull. Math. Biol. (1998) 60: 1123. https://doi.org/10.1006/S0092-8240(98)90005-2

Zhu, K., & Ying, L. (2014). Information Source Detection in the SIR Model: A Sample-Path-Based Approach. IEEE/ACM Transactions on Networking. DOI: 10.1109/TNET.2014.2364972