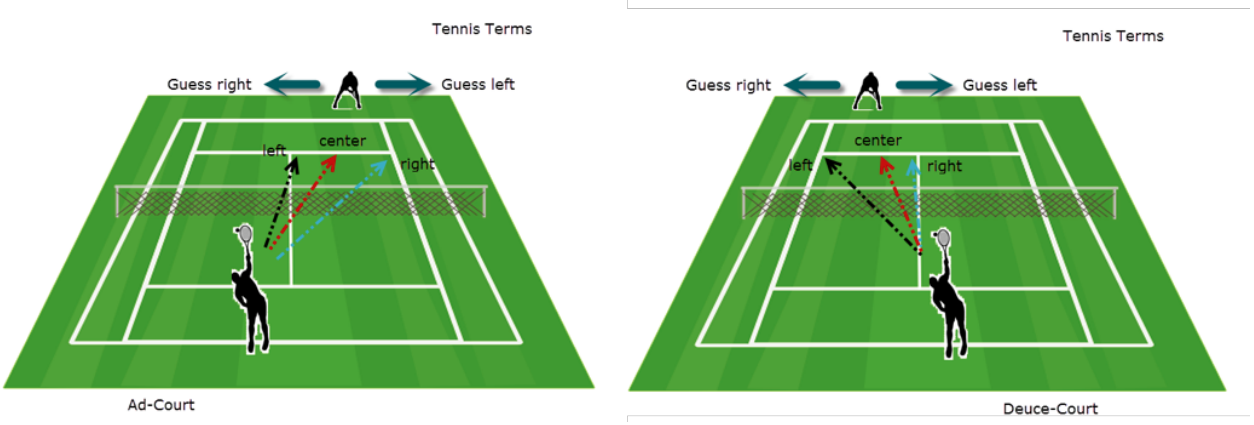
Group Project Part #1: Game Theory

Question
Do professional tennis players serve to each direction at an optimal equilibrium proportion according to mixed strategy game theory?

Analysis of the Problem
Game theory is a branch of modeling that analyzes decision-making behavior of multiple parties. In one of its basic forms, game theory describes a total conflict game between two persons where one 'wins' at the expense of the other; therefore, the total expected outcome sums to 100%.

Tennis may offer an opportunity to examine two-person total conflict game theory in real life. In particular, each point begins with a server serving the ball to a returner in one of three directions: left, center, or right (see Figure 1). The returner then also guesses which of these directions the serve will go. We assume there is no dominant (pure) strategy or else the server would only ever serve one way and the returner guess one way. In observing real tennis matches, this is not the case because if the server always served left, the returner would anticipate this and be better prepared to hit a stronger return. Similarly, if the returner always leaned to one direction, the server would serve the opposite way. As a result, both the server and returner have to vary their direction in order to maximize their probability of winning the point. In game theory, this is known as mixed strategy and at a stable equilibrium proportion, the probability of winning the point becomes independent of serve or return direction. Our project objective, therefore, is to analyze if mixed strategy equilibrium play actually happens during real tennis matches.

Figure 1: Tennis mixed strategy options

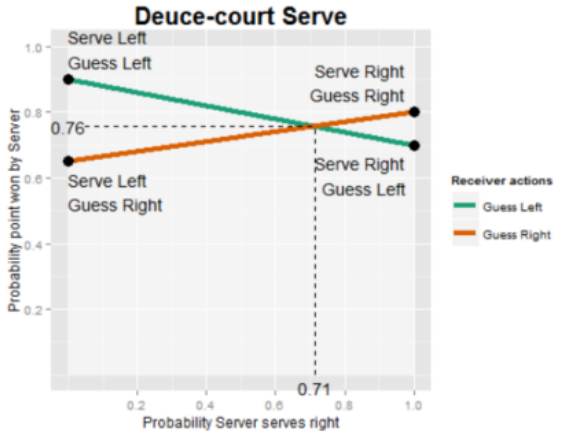


NOTICE: Serve right, Guess left (and Serve left, Guess right) means the receiver guessed correctly

Outcome Expectation

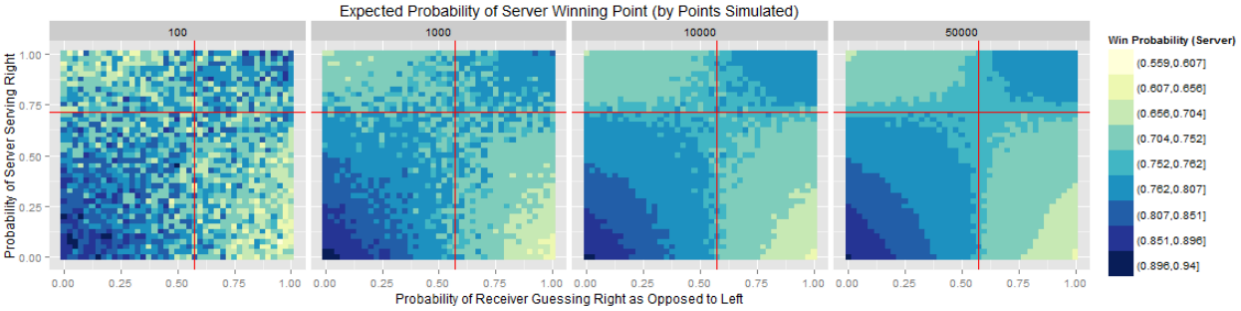
There are four scenarios in each match that must be considered individually: player 1 serves from the ad-court, player 1 serves from the deuce-court, player 2 ad, player 2 deuce. One such scenario is shown in Figure 2. Evidence of equilibrium play must be examined separately for each scenario. The question then becomes, what evidence can we look for to support our assumption of equilibrium play?

Figure 2: Mixed strategy equilibrium



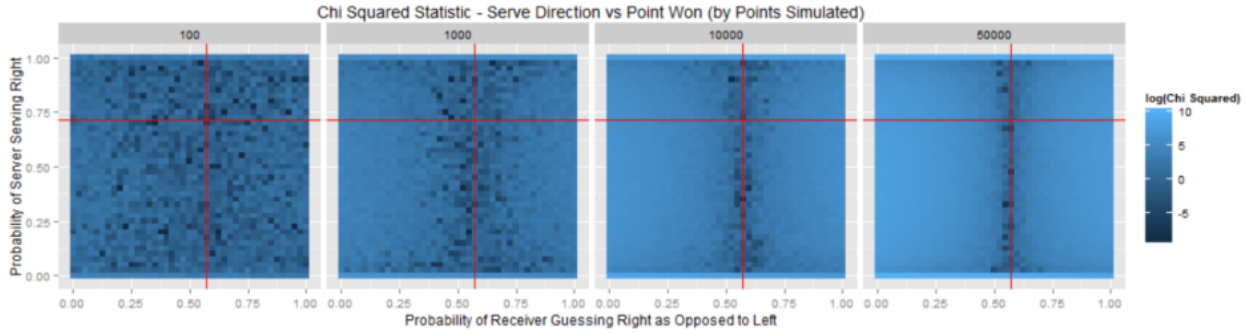
To answer this, let's look at a simulation of the game depicted in Figure 2.

Figure 3: Simulated game space



What we see in figure 3 is the expected value of the game for the server, for many combinations of serve/receive decisions. Given a simulation with enough games, it is easy to see the optimal values for the server and receiver of 71% and 57% respectively (shown by the red lines). However, we can also see that if we don't look at enough games we have a very noisy signal. Unfortunately, we can't observe the choice made by the receiver. We need another measure that we *can* calculate, that will show us when the play is in equilibrium. For that we turn to the Pearson Chi Squared Test.

Figure 4: Simulated chi-square tests



The Pearson Chi Squared Test is a measure of independence. We note that if the players are playing at equilibrium, then the proportion of time the server serves left and right should be independent of the number of points won to the left and to the right. This is convenient, since we can measure both quantities. Unfortunately, this is only true when the receiver is playing optimally, as shown in Figure 4. So we can use this method to determine the optimal play of the receiver, but not the server. The other problem with this method is that it is subject to a high number of false positives when the number of points is low. Since a typical match is less than 300 points, it is rare to have enough data to make a good estimate.

Figure 5

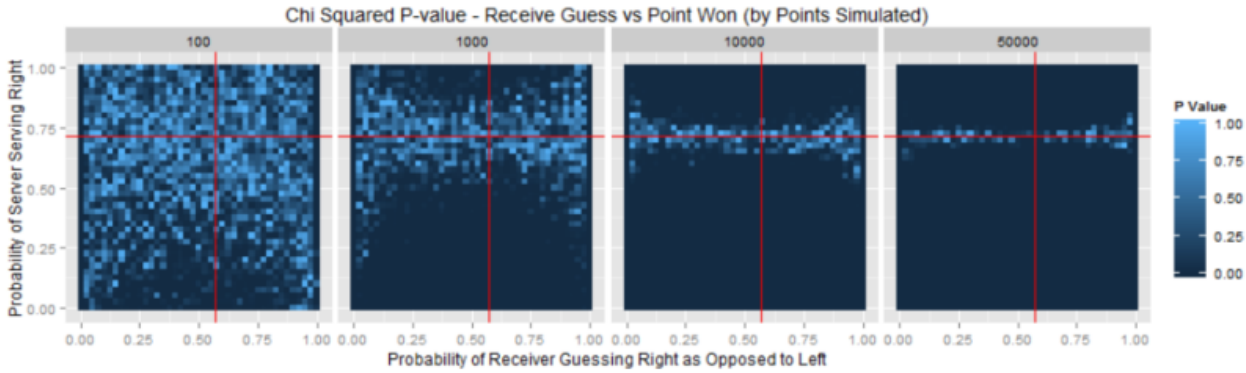


Figure 5 shows a very similar test. In this case, however, we are measuring the independence between the receivers choice of left vs right and the outcome of the point. Unfortunately, we can not use this for real world data since there is no way to know the receivers choice. It is also subject to the same issues with noise. Note that in this case, we are using the p-value from the test instead of the chi squared statistic, but that they work in much the same way.

End section with below paragraph, which leads into methodology and dataset?
The simulations show that equilibrium play only really becomes evident and very highly significant at 10,000 point simulations. In a well-contested tennis match, there is usually only approximately 300 points, which may be too few to observe true mixed strategy equilibrium play. Nevertheless, we are still interested in examining the possibility of mixed strategy game

theory occurring. Our expectation is that under the right match conditions, some evidence of mixed play equilibrium would be apparent but a lot of variance from this optimal point and potentially false positive results would be expected as well.

Methodology

Public databases for tennis matches and points are not overly detailed and certainly not in respect to service direction or service side. We, therefore, collected our own primary data using observation of full-length match video on YouTube. While it may seem prudent to aggregate random tennis matches to elevate total point sample size, tennis strategy is very dependent on player matchups, rankings, weather, court surface, tournament, and several other variables. As a result, we preferred to isolate our analysis to a single match, believing that despite a smaller sample size, these other match-specific variables would be mostly controlled and offer a truer representation of a game theory scenario.

The chosen match was a 2001 US Open quarterfinal between Pete Sampras and Andre Agassi (https://www.youtube.com/watch?v=ek3CqpKQo74). We reasoned this match was a strong candidate for increased chance of mixed play strategy due to the following characteristics:

- A high-stakes match in an important tournament would increase motivation to maximize strategy
- High-ranked players tend to have stronger serves and points won on serve
- Players very familiar with one another would know each other's tendencies better and try to capitalize on it
- The final match score was 6-7, 7-6, 7-6, 7-6 and no service breaks were made. which is a rare occurrence. This is a match where serves were particularly effective.
- The match was relatively long and close, offering more points for our sample

Data variables that were based off observation included:

- Server
- Side of court being played (deuce [right] or ad [left])
- Set and game number
- Serve number (1st or 2nd)
- Direction served to (left, center, right)
- Point won (binary yes/no)

Data cleaning and analysis were done using R (see Annex 1). Of note, serves to the center were ignored since they were difficult to accurately observe and generally uncommon (~1.5% of all serves). Only serves resulting in points were analyzed (i.e. service faults excluded). The chi-square test used the function chisq.test from the base R package.

Results

Sampras-Deuce

Serve	%Win Left	%Win Right	Chi-sq	p-val
1	82.1	74.1	0.158	0.691
2	76.9	73.3	0	1
Both	80.5	73.8	0.214	0.644

Sampras-Ad

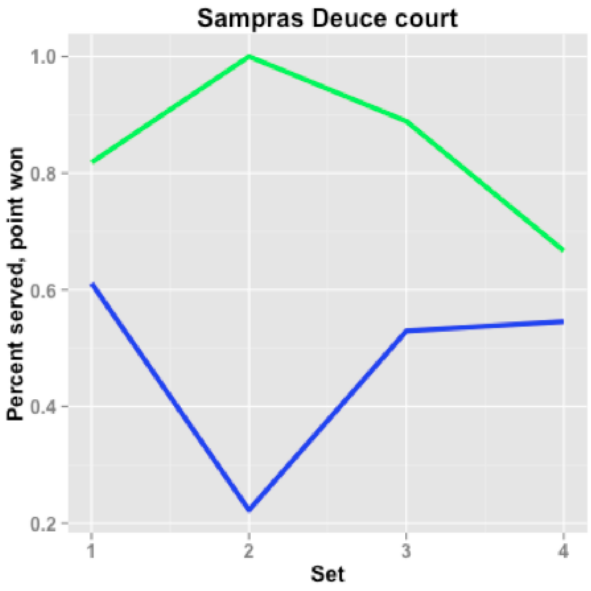
Serve	%Win Left	%Win Right	Chi-sq	p-val
1	75.0	85.2	0.253	0.615
2	58.8	71.4	0.013	0.908
Both	67.6	82.4	1.34	0.247

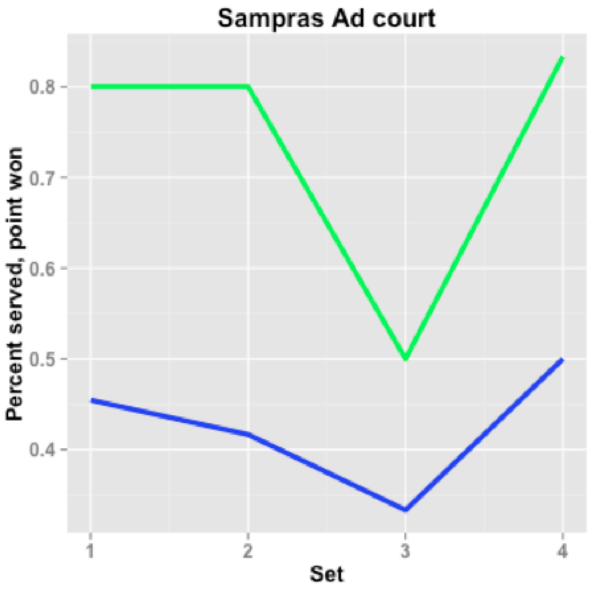
Agassi-Deuce

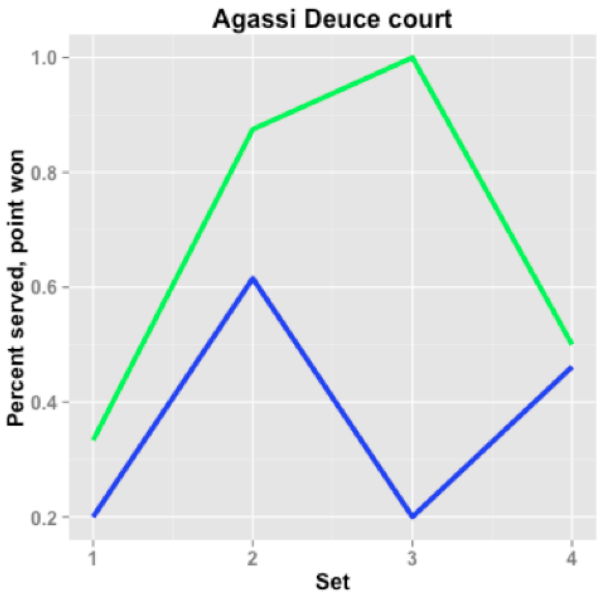
Serve	%Win Left	%Win Right	Chi-sq	p-val
1	70.0	83.3	0.681	0.409
2	80.0	54.5	0.293	0.588
Both	72.0	72.4	0	1

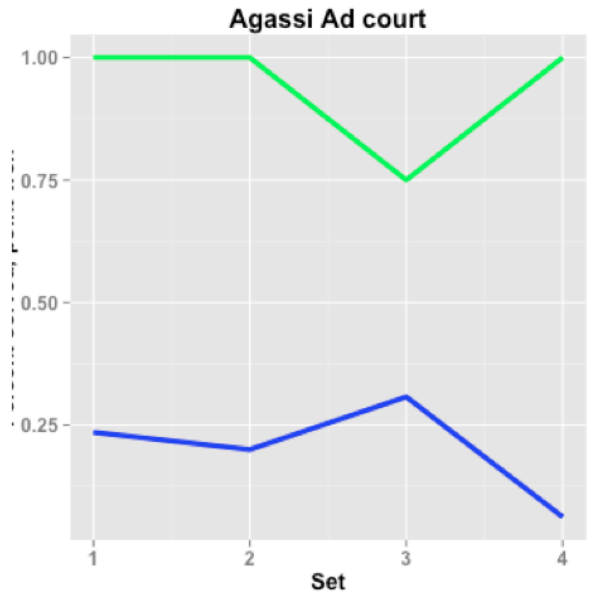
Agassi-Ad

Serve	%Win Left	%Win Right	Chi-sq	p-val
1	90.9	68.9	1.207	0.272
2	40.0	50.0	0.000	1.000
Both	75.0	64.4	0.248	0.618


Sampras Deuce court
Percent served, point won
1.0
0.8
0.6
0.4
0.2
1
2
3
4
Set


Sampras Ad court
Percent served, point won
0.8
0.7
0.6
0.5
0.4
1
2
3
4
Set


Agassi Deuce court
Percent served, point won
1.0
0.8
0.6
0.4
0.2
1
2
3
4
Set


Agassi Ad court
1.00
0.75
0.50
0.25
1
2
3
4
Set

Conclusion

The real match data showed some evidence of optimal mixed strategy if analyzing the significance value (p-value) of the chi-square statistic. The percentage of serves to each side, however, fluctuated during the course of the match whereas we may have expected to find a more stable equilibrium point throughout. This finding may be due to the players reacting and adjusting strategies, but is more likely that the smaller sample size (overall and within a set) results in wider confidence intervals and more variation as initially expected. In comparison with the simulation exercise, the added noise in our limited sample size of 100-125 points per server-side combination should not be overly surprising. Nevertheless, it is still telling that no matter how the data is subsetted, chi-square independence still holds and in many cases at quite large p-values.

In addition to sample size obstacles that any single match would have, the model is limited by the inability to accurately measure which side the returner is guessing. If this information were available, we could more directly estimate the optimal equilibrium proportion instead of indirectly determining if the observed proportion yields statistically similar outcomes. The model and analysis also naively assumes serve direction fully dictates whether a point is won or lost, which is only partially true in reality. A future extension of this project may look solely at points where the relationship between serve direction and point outcome is more obvious. It would also be interesting to compare results of this ‘high-potential game theory’ match to one ‘lower-potential’ candidate with reduced stakes and unfamiliar lesser-ranked players. Finally it is unknown to what extent players are actively aware what their service direction proportions are, let alone perform at some optimal equilibrium value. The fact there is even some evidence of optimum play observed lends support to professional tennis players as strategic players when it comes to directing and varying their service games.

Group Project Part #2 - Graph Theory

Question

How can we model relationships on Twitter using graph theory? Can we use graph theory to help us find important people within a community of interest?

Analysis of the Problem

Twitter is an online social networking service that enables users to send and read short 140-character messages called "tweets". Users of Twitter can connect to other users by 'following' one another, which allows you to see a stream of tweets from them. As a way to model the structure of Twitter, we can use graph theory to help model these relationships between people on Twitter.

Graph theory is a mathematical formalism for describing relationships between things. A graph G consists of two sets, a set of vertices (nodes) and a set of edges (links). An edge simply connects two vertices, which represents some kind of relationship between them. A vertex is normally labeled with a letter or some kind of identifier. Edges are labeled by the two vertices it connects. The degree of a vertex is simply the number of edges connecting to a vertex.

In our models going forward, we need to introduce two extra concepts to extend graph theory:

1. Using a property graph, we can add properties to nodes and edges. This will help us differentiate between different kinds of entities and relationships in our graphs.
2. Edges can have a directionality associated with them, so the connect from a source vertex to a destination vertex
    a. This allows us to define an in-degree, which is the number of edges that point to the vertex, not away from it.

We can use this formalism to help model Twitter, by using vertices to represent users and tweets (entities in the Twitter domain), and edges to represent relations between them, like ‘following’, ‘tweeted’ and ‘retweeted’.

Using this, we can define our problem more clearly: using the constructs above, can we 1) use data from Twitter to create a graph model of a community of interest and 2) can this model help find other important people within this community?

Methodology: Crawling Twitter

To proceed with part 1, we decided to map out the Data Science community on Twitter. To do this, the plan is to construct a graph model where nodes represent users and edges represent a follow relationship using the Twitter API.

We did this by starting with some known people in the Data Science world, like Hilary Mason, which we call *seed nodes*. We then used the Twitter API to find who follows these known people. These people are known as being *one degree of separation* away from the seed nodes. We also took these users and found out who they follow as well, getting us new nodes that are *two degrees of separation*.

Crawling twitter proved to have some challenges, due to:

- Twitter's API has a limitation on the number of API calls that can be made, so crawling is slow
- Need good error handling and backoff logic to handle errors from Twitter's API

To see the implementation of the crawl, please review twitter_crawl.py in the code repository. The output of this script are pickled objects that allow us to load the results into other scripts. An overview of the process is shown here:

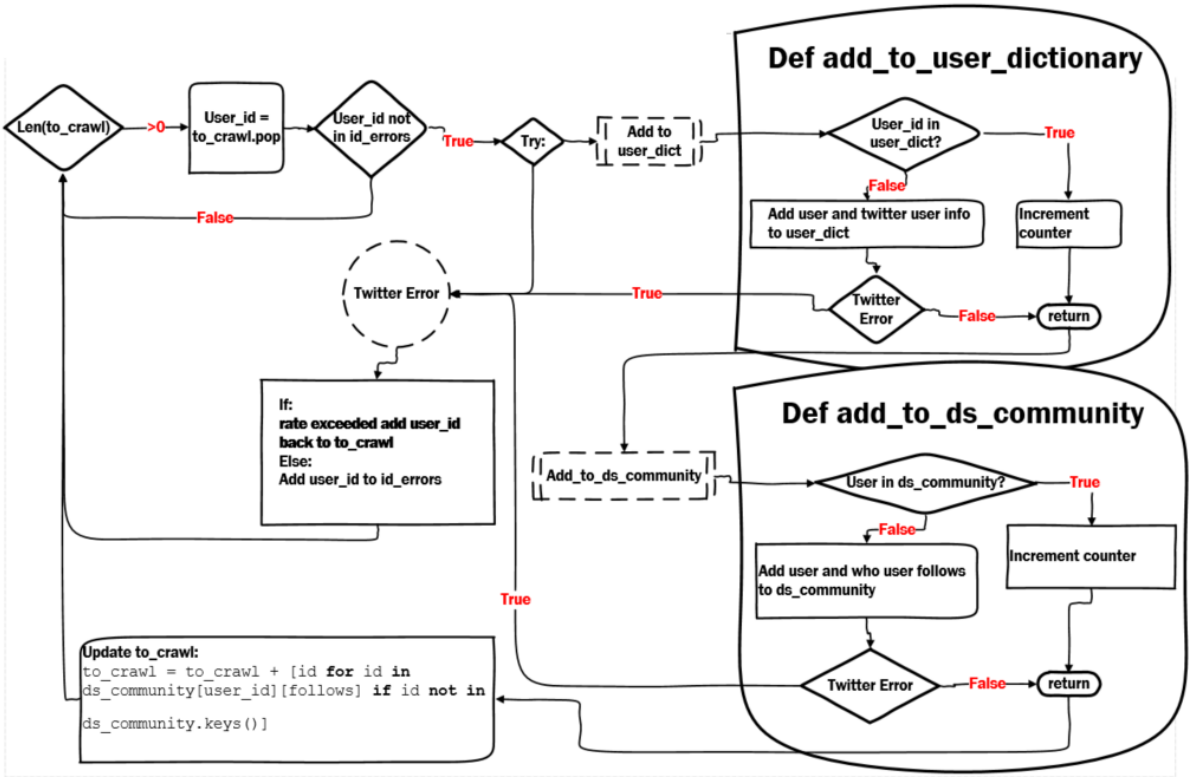


Figure 2.1 - Flowchart of Twitter Crawl Implementation

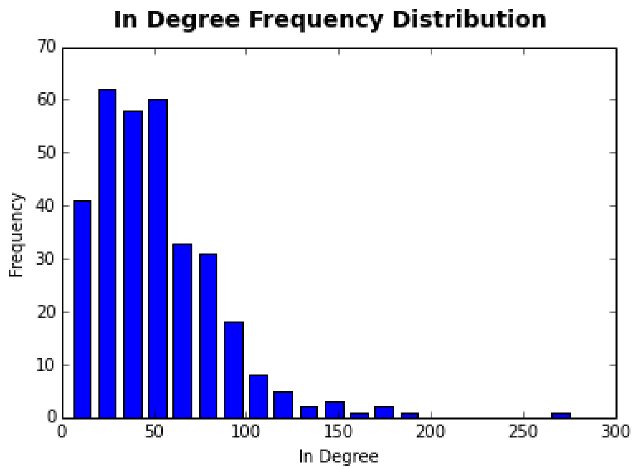
Methodology: Finding Important Nodes via Centrality

Once the crawl is finished, we can use the generated objects to construct a graph. This was done in Python via NetworkX. Our first step was to attempt to visualize the graph, but due to the high number of nodes and edges, was mostly unreadable. Instead, we need to find a way of categorizing node base don importance first, and then use that to layout the graph differently.

To find these important nodes, we can borrow a notion from graph theory on centrality. Centrality is a measure of how ‘central’ or ‘important’ a node is. This notion is a bit fuzzy and can be defined in many ways:

- **Degree Centrality** - One metric that can help define which vertices are important is using the number of links incident upon a node. The more edges on a node, the more 'important' it might be. This is one the simplest metrics for centrality.
- **Closeness Centrality** - In graph theory, the *shortest path* is the minimal amount of hops needed to travel between two nodes. We define another distance metric that is defined as the sum of its shortest path distances to all other nodes. The smaller the number, the closer this node is to all other nodes, so we can define *closeness* as the reciprocal of this metric. We will not use this metric.
- **Betweenness Centrality** - this metric quantifies the number of times a node acts as a 'bridge' along the shortest path between two other nodes. The more often a node is along shortest path in the graph, the more likely it is to be important. Notice how both closeness and betweenness are defined in terms of the whole graph, while degree centrality is defined locally per node.
- **PageRank** - is an iterative algorithm that outputs a probability distribution over nodes that represents the likelihood that a person randomly moving along edges will arrive at any particular node. After each iteration, the node's value is an approximate PageRank value that approaches the theoretical true value over time.

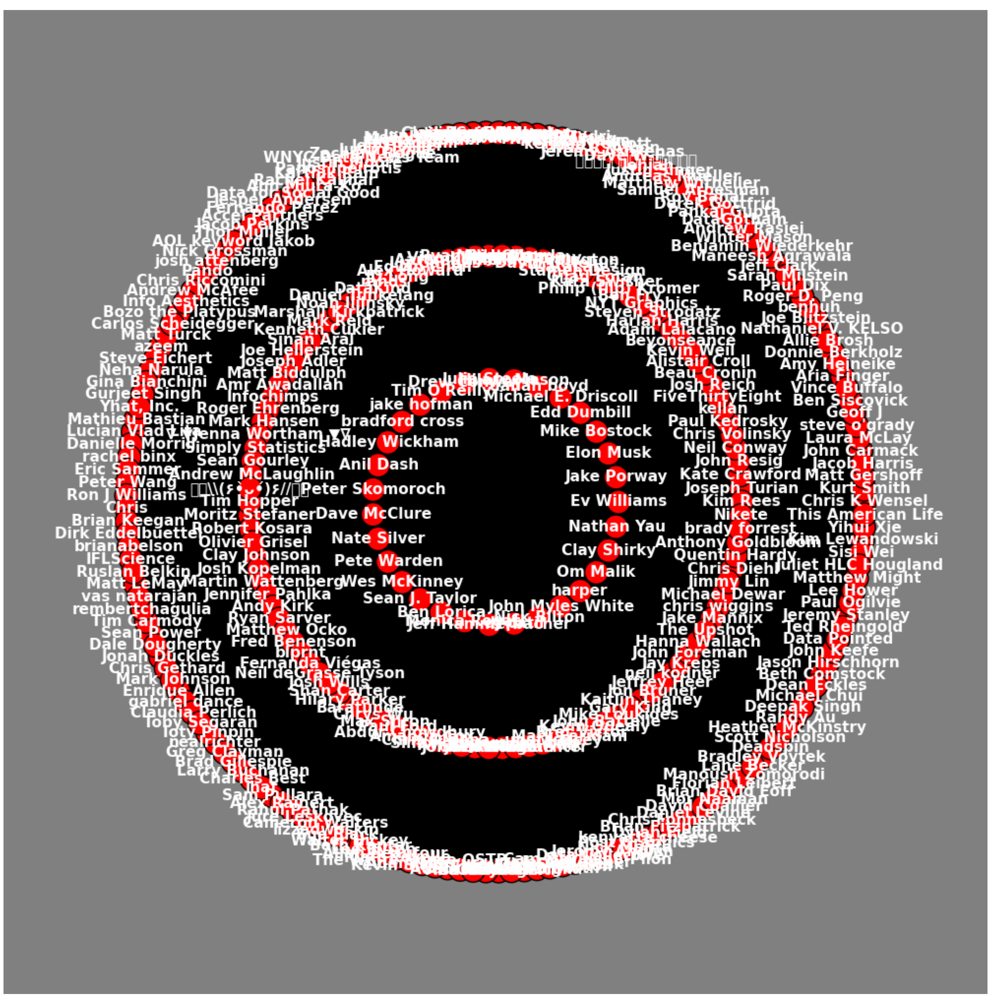
We decided to use degree centrality and find nodes that high the highest in-degree, and consider those nodes to central to the data science community. We can use our crawled graph and see if we can find nodes with a high number of edges coming in using networkx's in_degree() function. We can look at a histogram of in-degree values, which is shown below in **Figure 2.2**. Notice the long tail of nodes with large number of incoming edges; this is where we want to find the important nodes.



We can use this information to make a decision on how to layout the graph for visualizing. Separating the nodes into three groups based on their in-degree, we can visualize each group as a shell. Figure 2.3 is the final visual output for the crawled results, using the in-degree information to help layout graph, with nodes in the inner shell more 'central' or 'important' than others.

Figure 2.2 - In Degree distribution over crawled graph

Figure 2.3 - Trimmed Graph of Data Science Twitter Crawl



Looking at nodes with an in-degree > 100, we found these nodes that were not a part of the seed group we used to start crawling (the code for this is in network_graph.py). The listing of central nodes found here is in the **Results** section as **Listing 2.1.**

Note, that we also tried using PageRank to find more central nodes (code pagerank.py), but the results matched the results from the degree centrality methodology, so they have been omitted.

Methodology: Influence and Reach

We attempted to find important nodes using another metric. On social media, influence is some kind of measure of how much influence you have over the behavior of other users on the network. One feature that is used in calculating influence, by services like Klout, is ‘reach’, or how many users see content posted by a user.

To calculate reach for a given tweet made by a user, our methodology is as follows: for that given tweet, find a list of users who retweeted the tweet, and for each user who retweeted, get a list of their followers. Calculating the total number of in-degree links on these users gives us an upper bound on the number of people who saw a tweet in their stream. Due to limitations, like twitter’s API rate limiting, and laptop capability, we can only show a resulting graph for a single tweet. Here is a very small snippet of a graph built from an example tweet from Hilary Mason; the node in the middle is a tweet; the nodes connecting to it are the users who retweeted it, and the outer shell are followers of those users.

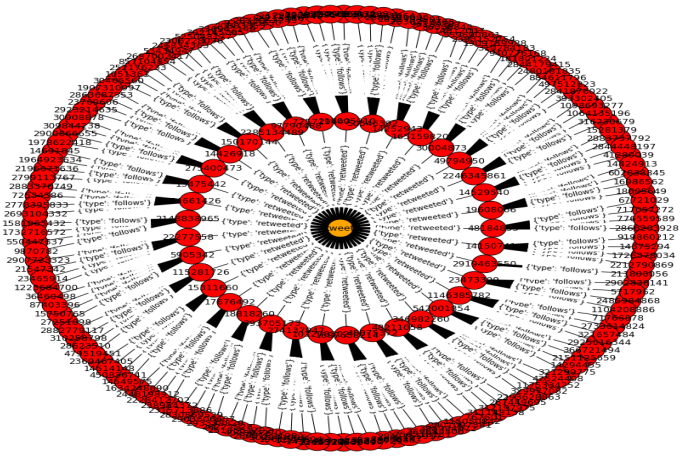


Figure 2.4 - Graph Model for Calculating Reach Metric

Results:

Here is a listing of the central nodes found via the degree centrality metric:

- **Ev Williams** - co-founder of twitter
- **Jake Porway** - Founder and Executive Director @ DataKind
- **Elon Musk** - Not really in DS community but in broader tech community
- **Edd Dumbill**, VP Strategy at Silicon Valley Data Science
- **Michael E. Driscoll** - Founder + CEO @Metamarkets; data; analytics; visualization
- **danah boyd** - Microsoft Researcher; focuses on everyday practices involving social media
- **Julie Steele** - Director of Communications at Silicon Valley Data Science
- **Tim O'Reilly** - Founder of O'Reilly Media and a supporter of the free / open source movements
- **jake hofman** - research scientist at msr nyc, interested in learning from data.
- **bradford cross** - Founder CEO focused on consumer products and data technologies.
- **Anil Dash** - CoFounder CEO of ThinkUp, the first analytics tool for social media
- **Peter Skomoroch** - data scientist @ Data Collective
- **Dave McClure** - Entrepreneur and prominent angel investor
- **Pete Warden**- Engineer at Google
- **Wes McKinney** - Author of "Python for Data Analysis"
- **Sean J. Taylor** - Social Scientist. Hacker. Facebook Data Science Team
- **Ben Lorica** - Chief Data Scientist at O'Reilly Media, Inc
- **Monica Rogati** - Data scientist with a passion for turning data into products
- **Jeff Hammerbacher** - Founder and Chief Scientist of Cloudera
- **Nick Bilton** - Journalist; formerly lead blogger for the New York Times' bits blog
- **John Myles White** - Authors of Machine Learning for Hackers
- **harper** - Former CTO @ Obama for America; founder of Modest
- **Om Malik** - Web and technology writer; founder of and senior writer for GigaOM.
- **Clay Shirky** - Writer on the social and economic effects of Internet technologies
- **Nathan Yau** - Owner of FlowingData. Author of ‘Visualize This’

Listing 2.1: List of Central Nodes in Data Science Community Based on Degree Centrality

The vast majority of them seem to be good candidates for being in the Data Science community. Notice that some users are not really in the data science community, like Clay Shirky, Dave McClure and Elon Musk. This highlights another aspect of graph theory, and that is of *communities* and *community detection*. Communities are groups of nodes in a graph that are highly connected, indicated that they 'belong' together for some reason.

For example, we have tried to crawl some part of the data science community with our crawl above. However, in graphs, communities can overlap, where nodes can belong to more than one group. In this case, Elon Musk is followed by a lot of people in the DS community, even though one wouldn't consider him to be a member.

Conclusion / Improvements

Crawling twitter from their API and creating a graph model from it proved to be helpful in finding central nodes to a community.

Here is a list of improvements that can be made for future projects:

- Expand the crawl to gather more data from Twitter and cluster it to find communities of interest. Due to the API rate limiting, this might have to be clustered
- Use different metrics for centrality to see if there are important differences.
- Expand on the influence calculation, and compare users based on an expanded reach metric (calculate average reach based on latest tweets, use it to compare nodes in a network)