Summary Report: Capstone 2 Project
Social Network Outreach Analysis
Springboard Data Science Career Track

Eric Cruz

Introduction

The title of this report reflects a potentially large scope, as the report was developed from a project proposal with a problem statement and general outline of how to solve the problem. As this report is intended to be shared as part of the deliverable to a client base, I would encourage interested clients to review the proposal document. It contains ideas about analyzing trends on various social media platforms such as LinkedIn, Facebook, Instagram, Twitter, and the like. It also provides some background on the context of the initial dataset, which is a curated and exhaustive outreach list for the Cornell Women's Tennis program.

As this project developed, the scope was reduced very quickly, as it became clear that even if you could gather data at will, sifting through it is a much larger and difficult task. The scope of this report will focus on describing a process that was developed to produce a deliverable, which can potentially be repeated at scale. The deliverable is a report accompanied by a dataset which contains the roster list for a given college tennis program, the result of a LinkedIn search for the people on the roster, and the prediction from a model indicating whether the search result has found the right person. The dataset containing a number of interesting fields from the rosters and LinkedIn will also be provided, and is intended to help the client make the final determination of whether the LinkedIn search was successful. Many searches did not turn up any result, and others have a result which is not the right person. The use of a binary classification model can be taken with a grain of salt, but I will speculate on what is really happening behind the scenes. Since the model is built by starting with a set of labelled data, or the answer key of whether the LinkedIn profile is the right person, the predictions mimic the likelihood as determined by the verified data. For example, there are people who share the same name, and the person on the roster does not have a LinkedIn account. It is therefore impossible to find the accurate search result, but likely there will be someone on LinkedIn with the same name who is not the person on the roster. Let's just say that the "wrong" person also has most or even all of the characteristics that usually mean we have found the "right" person. The model predictions will reflect the fact that even when a particular observation looks like it should be right, the true result is that the person can not be found. Therefore, the model must make some amount of predictions that a matching name is the wrong person. This example is an oversimplification, but I wanted to give some cover to model predictions that will inevitably contain false positive and false negative results. Having the dataset with all the features used in

the model will help the client eyeball which predictions are suspicious and prioritize which results should be double checked.

Problem Statement

The Intercollegiate Tennis Association (ITA) is the governing body of college tennis, overseeing men's and women's varsity tennis at all levels - NCAA Divisions I, II and III, NAIA and Junior/Community College. The College Tennis Alumni Network (CTAN) was formed under the ITA, and already has chapters in several American cities as well as Australia. Each chapter is a volunteer-led group helping former college tennis players in the area connect through events and other resources. The problem statement is to develop a resource which can be shared by all existing chapters and potential new chapters, as a tool for conducting an outreach campaign. That resource is a procedure for gathering information from college website rosters and letter winner lists, conducting a LinkedIn search by name and school, producing a report which predicts the accuracy of the search results, and providing it along with the resulting data. The report will be provided to volunteers who can use it to contact Alumni from their school.

Clients

The clients are the volunteers who will be provided with the data and report, and other interested parties such as coaches and college program supporters. To develop a preliminary set of target clients, Corey Pegram from CTAN provided me with the list of existing members with a New York State home address, with the specific intention of asking for volunteers to form the New York chapter and offering to provide them with the data and report for their school. There are 142 members, approximately evenly split between Men and Women, representing 88 different schools. In general, most colleges have online rosters going back to the mid 2000’s, with some going back much further. Most websites also have a document available for download, which has a list of All-time letter winners.

Approach

The approach was to start by downloading rosters from college websites. Starting out as a novice, I first evaluated some commercially available web scraping tools, hoping to be able to search Facebook, Instagram, Twitter, and LinkedIn. There are many tools for purchase which advertise being able to search those types of websites, but they are so bad it isn’t worth describing their limitations. I will briefly describe the journey that led me to select the tools I chose while describing how the data was gathered. I use a free version of Octoparse web scraping software to cycle through all the roster years available on the college websites and download them. I use Selenium with a Google Chrome extension to write a Python program in a Jupyter Notebook for the LinkedIn search and download. I leverage my Capstone Project 1 on Fraud Detection to select a binary classification model and perform feature selection for the model. The model chosen is a Random Forest Classifier, which also won out in that project.

Exploratory Data Analysis (EDA)

Obtaining the Data

The deliverable datasets are the set of downloads for a College Tennis Team's historical rosters from the school's website, matched with a search for the roster name/school combination on LinkedIn. The roster download is obtained using a web scraping tool called Octoparse, which has a free version that is adequate for obtaining rosters from the scope of nearly 200 teams in the original New York outreach campaign. This represents nearly 100 schools, for both a Men's and Women's team. The LinkedIn search is obtained using Selenium WebDriver with ChromeDriver, and writing a Python script in a Jupyter Notebook.

Octoparse gui-based web scraping tool

- The tutorial for scraping elements from a web page is straightforward and simple, as just clicking on what you want to scrape created the script behind the scenes. The slightly more complex part is figuring out how to cycle through the available roster years, usually from a drop-down list.
- The initial collection included about 60 different teams, or 30 schools for both Men and Women. Most schools use one of 3 or 4 different formats for organizing the archived rosters.
- There was some quasi-scripting involved for a few of the formats, such as creating a list of roster years in Excel based on a pattern of switching a few numbers to represent different years. Those involve defining the set of years to scrape and identifying patterns in the web address organization, but a large portion of websites are organized so all the available years are cycled through automatically.
- The fields chosen for download are Name, School, Year (e.g Freshman, Senior), Home Town, Roster Identification (e.g. Men's 2015 Roster), and the profile Link.

Selenium Python Script in Jupyter Notebook for LinkedIn Search

- A few different tutorials can be found online with a search engine such as Google, which follow a pattern of using Selenium to log in to LinkedIn with your own account credentials, then performing a keyword search in a search engine such as Google or Bing, and using the profile URL to scrape details from LinkedIn
- A tutorial which uses the Bing API was the easiest to implement, and offered a free trial with limited use keys for evaluating the API product. Aside from the cost of purchasing a tier-based subscription, the result was very disappointing with respect to the Bing Search. As pointed out, it seems ironic that Bing is a Microsoft product, and LinkedIn is also owned by Microsoft and tries to make it difficult to scrape their data. In short, the Bing search started returning different results using the same keywords, with worse results for finding the target person each time. That seems suspicious given the association of both products with Microsoft.

  https://codeburst.io/bulk-researching-linkedin-contacts-with-python-microsofts-bing-api-1f9a19e1c7b7

- The tutorial most useful, in my opinion, uses an ipython terminal which is useful for

learning about detecting elements for scraping, but it was not hard to convert to using a Jupyter notebook, which is more in line with the Springboard course curriculum.

https://www.linkedin.com/pulse/how-easy-scraping-data-from-linkedin-profiles-david-craven/

- Using a Google search instead of a Bing search, which does not require an API for purchase, better results were produced on a known keyword search. However, the search was still a disappointment knowing that the LinkedIn URL string was used in the keyword and the intended result was still less accurate than expected. However, I came to realize that the reason the tutorials use a search engine is they are looking for profiles with characteristics, not a specific person. I was able to use Selenium to make a direct keyword search in LinkedIn, and collect the URL from that search for evaluation of accuracy.
- The keyword search used is the “Name + School”, for example to search myself the keyword is “Eric Cruz Cornell”. There are some online discussions regarding the LinkedIn Terms of Service with respect to volume and crawling, so I proceeded cautiously and applied for a “whitelist exemption” for permission to crawl. A response has not been provided after about a month, but it took just over 2 months to receive an official denial of my request to obtain the LinkedIn Marketing API toolkit which includes the PeopleSearch API.
- To summarize the LinkedIn data collection, the base test cases for the Cornell Men’s and Women’s tennis teams ran a search on over 250 names each, and the fields chosen to collect are Name, Profile URL, Location, Company, School, and Activities.

Data Wrangling

Roster Data

The roster page usually includes a headshot photo, which downloads as a player profile link along with a separate label containing the person’s name. There is also a field containing the name, but when there is no photo, it is defined as a different field. In the years later than 2001, there are only a few missing photos. After the gap where the data resumes for 1963 back to 1947, there are no photos. The name columns can be combined using the fillna() function in order to get all the names in a single column with no missing values. After updating the column names with meaningful descriptions, the roster data contains the following fields:

Name, RosterYear, ClassYear, Hometown, HS, RosterLink

The ClassYear field identifies year in school such as Freshman, Sophomore, etc. The RosterYear identifies the web page such as “2001-02 Men’s Tennis Roster”. Some school websites include the school name in this label, but others don’t. For a person who is on the roster for multiple years, the dataset will have duplicate rows with respect to the Name, Hometown, and HS. The RosterLink appears to be unique for each year, but when clicking the link, the website only renders the most recent from those belonging to the same person.

Roster Data Transformations

groupby() aggregation : unique, max, count, first, last

In order to collapse multiple observations for the same name, we can use the groupby() function with aggregation “statistics”. For RosterYear and ClassYear, we select ‘unique’ to get a list of

the values as a cell in each row under that column. For example, a typical 4-year player would have an aggregated ClassYear list of [Freshman, Sophomore, Junior, Senior]. Note that for some schools ClassYear values could include Graduate Student, or Red-shirt. We can use the aggregation ‘count’ statistic of any column to obtain a value for YearsPlayed, and ‘max’ to select only one value for the ClassYear field, and ‘last’ to obtain a single value in each column for the remaining fields.

Regex, regular expression text splitting

We can use the regex, or Regular Expression, module in Python to extract the numerical value for the year in RosterYear such as “2001-02 Men’s Tennis Roster” into a separate column. We split on the dash (-) and take the preceding text converted to an integer to obtain 2001. We add 1 to that value to obtain the year pertaining to the Spring season, or the value after the dash which we discarded, as a proxy for graduation year if a Senior year season is present. Although the majority of players do not play all four years, we can obtain the ‘max’ value from the groupby aggregation to approximate graduation year. We could further extract the maximum ClassYear and calculate the projected graduation year if not already included, but will not do that in the preliminary analysis. This is for consistency when appending the gap roster years from a pdf file of All-Time Letterwinners. That list only has the name and years a varsity letter was earned.

Nameparser utility function HumanName

The Name obtained from the roster includes the First and Last name as a single string, and for the years 1963 and earlier includes a Middle Initial. The preliminary LinkedIn search performed poorly when including a Middle Initial. Rather than strip it out, it could prove useful to parse it into a separate column for First Name, Last Name, and Middle Initial. Further, there are a handful of Middle Name values that are meaningful, especially for people from countries where the Middle Name is more like part of an extended Last Name. In addition, it is useful to have flexibility to substitute other name types such as Maiden Name vs. Married Name in the search string. The following elements are returned from the HumanName() function, which was used by converting the Name column to a list, and passing each value to a for loop. We can call the elements individually to keep or process them individually.

<HumanName : [title: ’’ first: '' middle: '' last: '' suffix: "nickname: ''"]>

In order to distinguish Middle Initials from Middle Names and Initials that precede a Middle Name, we can first strip the period and count the characters to make sure the count is 1. Otherwise, we might mistake a 2-character name for an Initial with a period. We can create a separate column for Middle Initial and Middle Name, such that only one or the other will be populated after checking the character count.

Roster Data Additions

An additional column was added to identify the school, which will be useful for merging the data from different schools into a consolidated database. Also, a column called Criteria was defined which is used to concatenate with the Name to form the LinkedIn search string. It was used as a short form of College, e.g. “Cornell” instead of “Cornell University”, but the field could contain any value. Further, a Gender classification is added while building the roster list, for the purpose of later combining lists for Men, Women, and different schools.

The All-Time Letter Winner list in pdf format was copied into Excel manually, and wrangled in

order to calculate the latest year a letter was earned, and the number of years a letter was earned. The names were separated into FirstName and LastName to go along with the names obtained in FirstLast format. The list was checked against the roster download so only names missing from the list were merged into the full roster list.

The full list of 16 columns resulting from the roster wrangling are listed here:

FirstLast, RosterYear, ClassYear, Hometown, HS, RosterLink, Year, Name, First, Last, MidInit, MidName, YearsPlayed, College, Criteria, and Gender

LinkedIn Data

The output of the Roster Data Wrangling step is a list of names and criteria to search with a LinkedIn query. The lists are used in a for loop to simulate the following activity, which is defined in the commercial use limits published at the link below:

https://www.linkedin.com/help/linkedin/answer/52950/commercial-use-limit?lang=en

- Searching profiles by name using the search box located at the top of every page on LinkedIn.com

For example, the base case for testing is my own Name & Criteria “Eric Cruz Cornell”. The data download includes Headline, Location, Company, School, and Activities. Only the current employer name is extracted, but for School and Activities, the full list is returned because people don’t necessarily list their undergraduate school first and sometimes not at all, and same for activities with the word of interest being “tennis”. The profile link is downloaded as “SearchResult” and the name associated with the search is labelled “LName”. For the purpose of indexing, the name used in the search is returned from the for loop in list format, and has to be converted back as part of the data wrangling procedures. To summarize, the list of columns extracted from the search are:

Name, LName, SearchResult, Headline, Location, Company, School, and Activities

Note that many of the searches do not return a result, and many others return a result with an altogether different name. This will be discussed in the exploratory data analysis.

LinkedIn Data Transformations

While building the model, an iterative feature selection process was used, which will be described in the exploratory data analysis section. For the purpose of building the datasets to be run through the model, the final selection of features is built as a Data Wrangling Phase on the LinkedIn data, so it can be passed along to the model to obtain a prediction of whether the SearchResult is considered a good match or not.

Boolean Values (Yes/No Indicators)

The following features are built from boolean values converted to integers based on the the LinkedIn data collected:

- School_yn - This indicates whether the LinkedIn profile includes the school. Since the search query criteria used is the school name, it seems likely that the profile found would include the school. However, there are a surprising number of verified search results

where the person does not include the undergraduate school where they played on the tennis team. Also, even for a search result that returned a different name, several profiles did not include the school used in the search criteria
- tennis_yn - This indicates whether “tennis” was listed in the Activities section, as you might expect many players to identify their participation on the team.
- SameTennis_yn - This feature is built by adding the 0,1 value of the School_yn and tennis_yn features, where 1 indicates True for those features. This measure is designed to give additional weight to profiles which identify both or neither of those features
- SameName_yn - This indicates whether the name associated with the LinkedIn search result is exactly the same as the input name.

Fuzzy Wuzzy Ratios (name match score)

As many of the names associated with the profiles found are similar but not exact, the FuzzyWuzzy python library was used to calculate how closely the returned name string matches the search name. This is a quick way to distinguish slightly different spellings such as Will vs William, or Pete vs Peter. There are 5 different ratios available, and all were tested. Rather than explain the subtle differences between the matching methods, all were kept in the final feature selection for reasons explained in the feature selection analysis. The names are: ratio, partial_ratio, token_sort_ratio, token_set_ratio, and WRatio. The result for each is returned as a value between 0 and 100, which is the equivalent of a percentage score on that scale.

At this stage, we merge the Roster data with the enhanced LinkedIn features and include an additional feature “Decade” calculated as a function of the “Year”, to complete the set of features selected for the model.

Test Parameters and Attributes

Model Selection

Predicting the accuracy of the search result means providing a yes/no answer to the question of whether or not the LinkedIn profile returned by the search is really the searched person.
Since this is a binary classification problem, there are several Machine Learning Models which are usually suggested as good candidates for this type of problem. The nature of the models is briefly introduced in this section.

Supervised Models

LR - LogisticRegression()

The Logistic Regression model is often mentioned as a good candidate for binary classification. It is similar to the Linear Regression model but predicts an output of true or false.

KNN - KNeighborsClassifier()

The KNeighborsClassifier is a supervised learning technique that learns from the labelled inputs to form classification clusters. The k represents the number of points, or “nearest neighbors” to consider when forming the clusters.

Ensemble Bagging

Ensembled algorithms are those which combine more than one algorithm of the same or different kind for classifying objects.

RFC - RandomForestClassifier()

Random Forest Classifier creates a set of decision trees from a randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object.

Preprocessing

The data was centered and normalized using the preprocessing function StandardScaler(). Correlation between all original, transformed, and created features was reviewed, and GridsearchCV (cross validation) was used to measure and compare feature importance using various combinations. The results were used to iteratively determine which features to include in the final model.

Numerical Data Types

Of the fields gathered and described above, 14 of the 34 fields are numeric, after converting boolean values to integers. Note that the model only uses numerical data, and we select those fields by type in order to define the X-value array for the model.

Define the binary categorical predictor y

The predictor, or y-value, was calculated for the Cornell Women dataset by simply comparing the manually researched LinkedIn profile URLs to those found by the search. For the Cornell Men dataset, the data extract was exported in csv format and opened in Excel. The URL links were reviewed, and the MatchLinkedIn column was inserted to mark the result. The updated dataset was read back into a pandas dataframe, and the Women and Men results were merged.

Train, Test, Split and Cross Validation

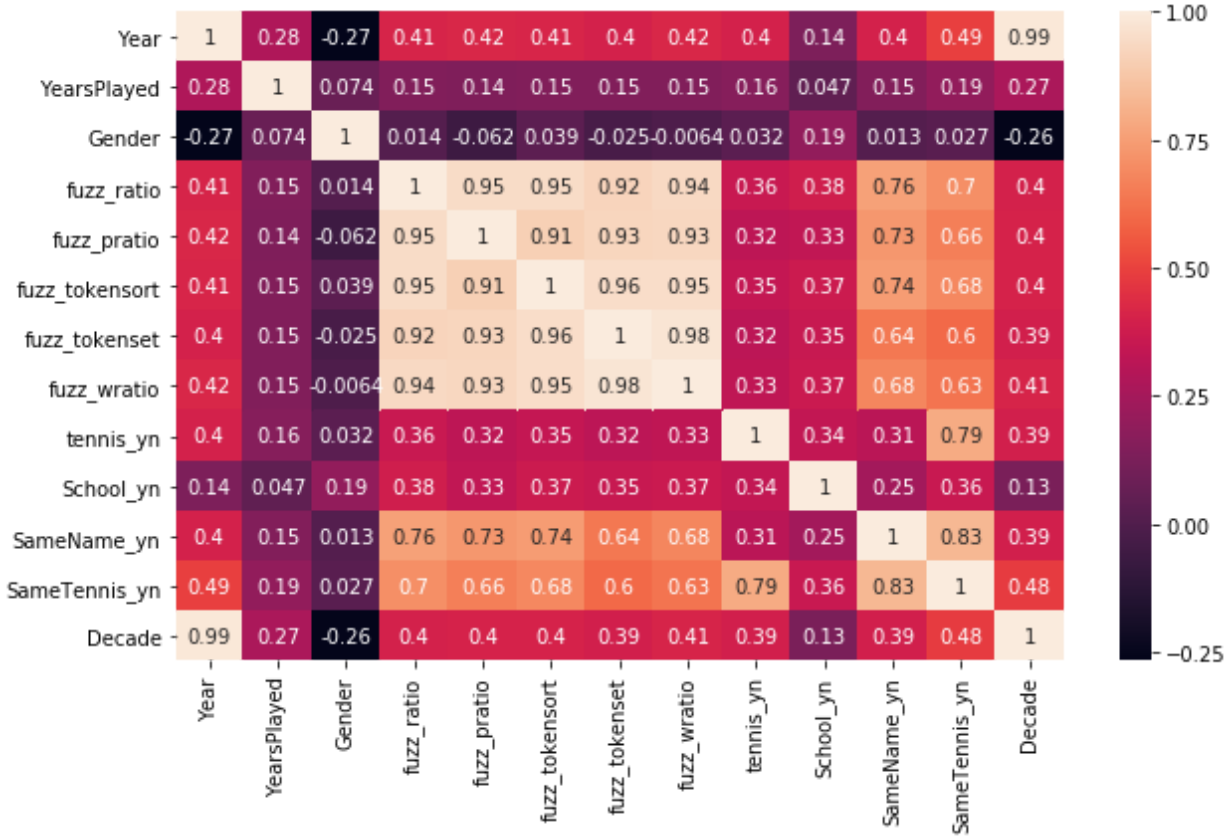
Per standard practice, the Train, Test, Split utility in sklearn was used in order to partition the dataset. This helps address the problems of overfitting, or learning the training data well without being able to generalize to the unseen test data, and underfitting, which is like not performing well even on the training data. The default split was used, so 75% of the data was used for training, and 25% for testing. Cross Validation was employed using GridSearchCV with default parameters. The default method is StratifiedKFold, a variation of KFold that returns stratified folds. The folds are made by preserving the percentage of samples for each class. In KFold Cross Validation, data is split into k different subsets (or folds). This uses k-1 subsets to train the data and leaves the last subset (or the last fold) as test data. Each fold is given an opportunity to be used as the holdout test set. The defaut of k=5 was used, so it is like simulating 5 times as much data.

Feature Selection

As this is a binary classification problem, we are not as interested in finding the most important features as we might be for a multiple class problem. Therefore, we are not as concerned with correlation between variables including those we construct from other features. Nevertheless, it is interesting to review how the feature importance shifts as we include the additional features with improved test results. As the Random Forest Classifier appears to be the best model in the baseline comparison, we use it to determine which features to include in our final model.

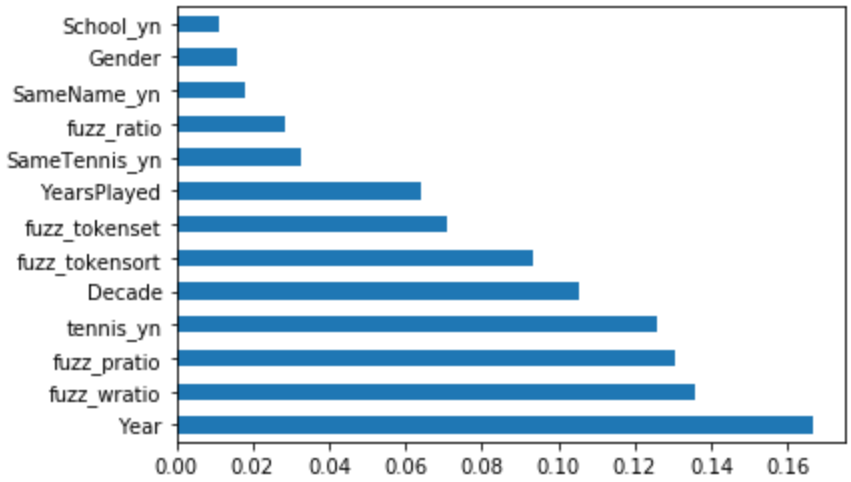
Correlation Check

The matrix with all the selected features is shown below. The features built as a function of one another are highly correlated, as well as all the fuzz ratios which are only slightly different in how they are calculated.



Feature Importance Visualization

We can use the feature importance attribute of the RFC ensemble model in sklearn to compare the relative scores. Several combinations were tested by eliminating some correlated features, but it did not have a significant impact on the results. Therefore, all the features were kept since they will be available in the download.



Parameter and Hyperparameter Selection

The model parameters, e.g. test split 25%, k=5 were mentioned above as using the defaults. Since the dataset used to build the model is relatively small, it does not take much computing time to let GridSearch try a variety of hyperparameters and automatically select the highest scoring. This did not have much impact on the results, so I am only listing the final selections here, and will not discuss the comparison. The Random Forest Classifier (RFC) model was chosen and only the following did not use the default.

- The default value model is indicated by criterion='gini'. Test results indicate that 'entropy' produces slightly better results.
- The default n_estimators=10 was changed to n_estimators=100 in a higher version. The new default will be used in the final model.
- The max_depth=None default will be changed to max_depth=30 based on GridsearchCV results of parameter tuning.

Measuring Model Performance

The original model selection was done using the Cornell Women data with 271 names. Since there was not much data, the test split was set to 70%, meaning 190 of 271 observations were used to train the model. This could lead to overfitting, but there is plenty of real world data to gather in order to verify the model results, and rebuild it with more data. The scores shown below are similar in terms of accuracy, but given the tradeoff between more false positives vs. false negatives, it is preferable to reduce the scenario where a profile is thought to be good but is actually the wrong person. The confusion matrix shown is formatted such that the selected model highlighted in green has 16 bad predictions for a person who was mistakenly thought to have the wrong profile found, and 12 bad predictions where the wrong profile was found but it was predicted to be right. The Random Forest Classifier (RFC), was chosen over Logistic Regression (LR), and K-Nearest Neighbors (KNN).

	[[72 12]	[[71 13]	[[68 16]
Confusion Matrix	[16 90]]	[19 87]]	[12 94]]
Model	LR	KNN	RFC
Accuracy score	0.8526	0.8316	0.8526
F1 score	0.8654	0.8447	0.8704
Precision score	0.8824	0.8700	0.8545
Recall score	0.8491	0.8208	0.8868

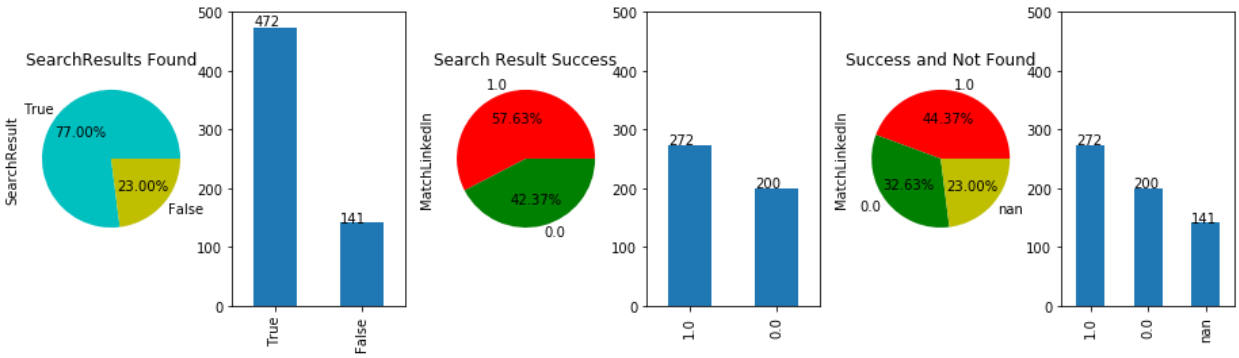
Test Results

Actual vs Predicted Result Summary

Actual Results, or Labelled Data used to build the Model

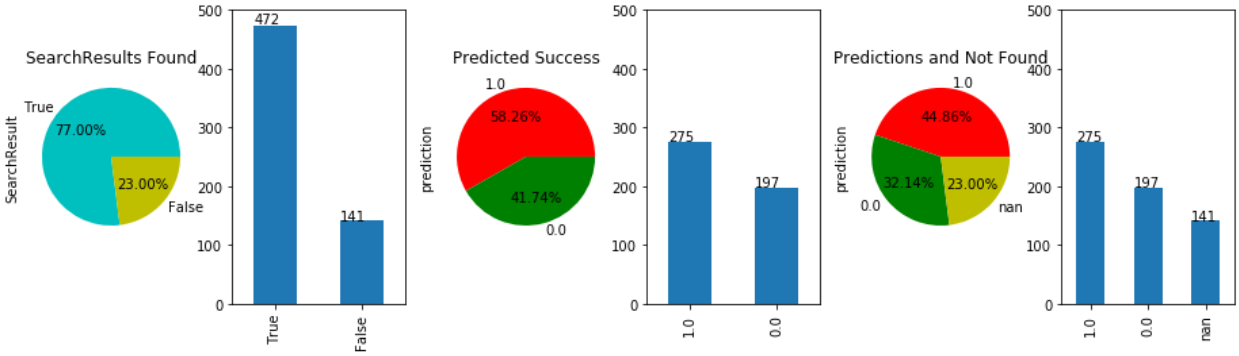
The summary of actual results is for the combined Cornell Men's and Women's Tennis Teams.

The number of LinkedIn profiles found is 472 out of 613 or 77.0%
The number of Matches for profiles found is 272 out of 472 or 57.63%
The overall ratio of Matching profiles is 272 out of 613 or 44.37%



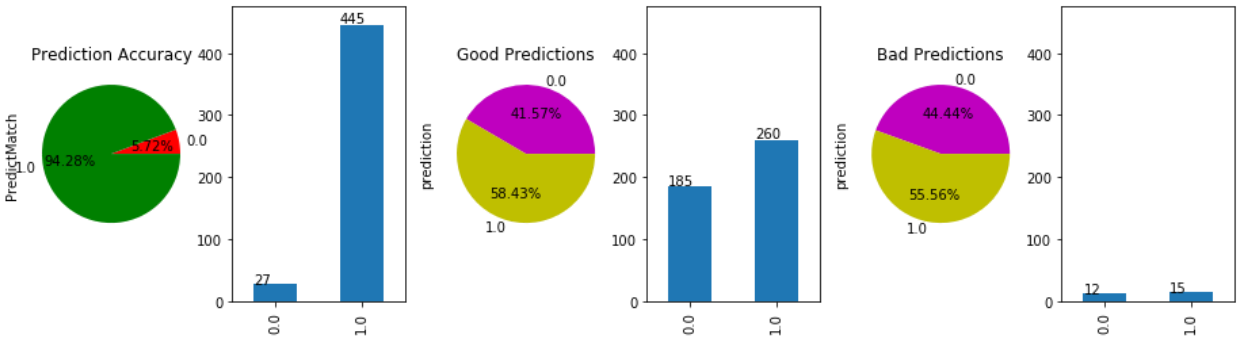
Predicted Results on the full dataset trained on 25% of the data

The number of LinkedIn profiles found is 472 out of 613 or 77.0%
The number of Predicted Matches for profiles found is 275 out of 472 or 57.63%
The number of Predicted Matching profiles is 275 out of 613 or 44.86%



Accuracy of Predicted Results

The number of Right Predictions is 445 out of 472 or 94.28%
The number of Right GoodMatch Predictions is 260 out of 275 or 94.55%
The number of Right NoMatch Predictions is 185 out of 197 or 93.91%
The number of Wrong GoodMatch Predictions is 15 out of 275 or 5.45% These are FALSE POSITIVES, or Wrong Profile is predicted as Right, and you could contact the wrong person
The number of Wrong NoMatch Predictions is 12 out of 197 or 6.09% These are FALSE NEGATIVES, or Right Profile is predicted as Wrong, so you could miss out on contacting some people



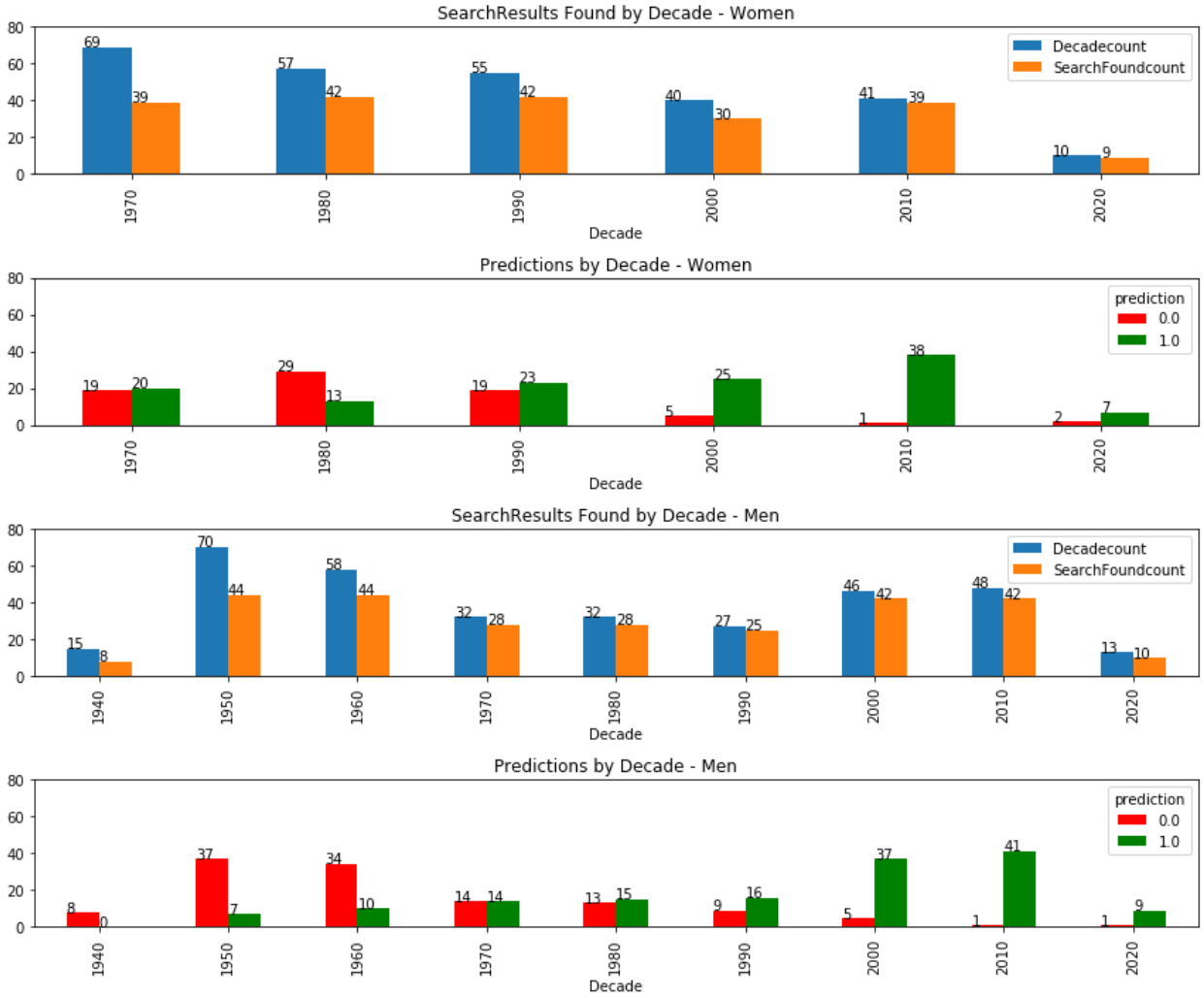
Trend Analysis

The trends will be analyzed visually to confirm the intuition behind selecting these features. Since the predictions were very accurate, the trends are consistent whether showing actual or predicted results.

Year and Decade - downward trend for older generations

The results for Women and Men are compared by restricting the Men's data to the same time period, only back to 1972. The trend shows predictions of Good Matches decreasing while going back in time.

The same data grouped by Decade is less busy to visualize, and the Men's data is shown going back to inception in 1947. Note the data prior to the 1970's is mostly "No Match" predictions.



Gender differences - Married vs. Maiden last names

Because the Men's data goes further back into a period with few good matches found, it skews the comparison between totals for Women vs. Men. On the other hand, the Women's data does not include adjustments for Married vs Maiden last names. The married names were provided by the University's alumni affairs office, and researched as part of the aforementioned outreach campaign. Before looking at the results grouped by gender, note the following statistics on maiden names for Cornell Women's Tennis:

- There are 105 married names out of 272 total Women, so 39% have a married name.
- There were 63 matching profiles found in the manual search for Women who have a married name on file, and 42 do not have a found profile on LinkedIn.
- Of the 63 found, 19 returned a bad LinkedIn search, 15 did not return anything from the search. This means 34 did not find the correct profile, and 29 found the correct profile.
- In most of the 29 found correctly, both the married and maiden name seem to be listed in the profile, or a married name is on file but not used on LinkedIn.
- There seem to be 4 good profiles found where the married name does not seem to be in the LinkedIn profile, and it is likely that they were found because the search history is retained on the account used in the automated search. Since the manual search for

profiles was performed before the automated search, the results are probably a little better than they would be for an account running the search for the first time.

- There are 50 profiles found manually for the master list of labelled data which were not found by the automated search. The 34 are included above in the count of married names, and there are another 16 which do not have a married name.

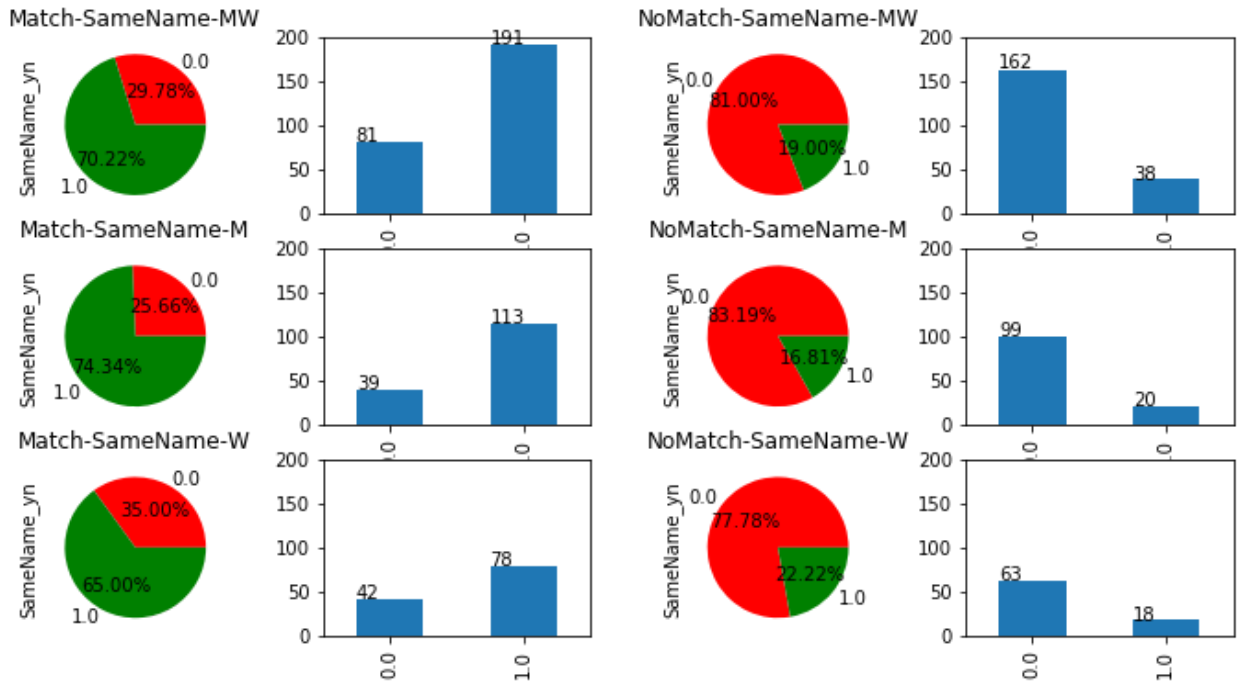
Although it seems more Women could be found by the automated search if married names were known, finding them is not part of the roster gathering process.

Boolean Features - confirming intuition

This section shows visualizations of the boolean feature distribution for Match vs NoMatch results, broken down by gender. The gender differences are not as pronounced as I would have expected, but still confirm the expected trends.

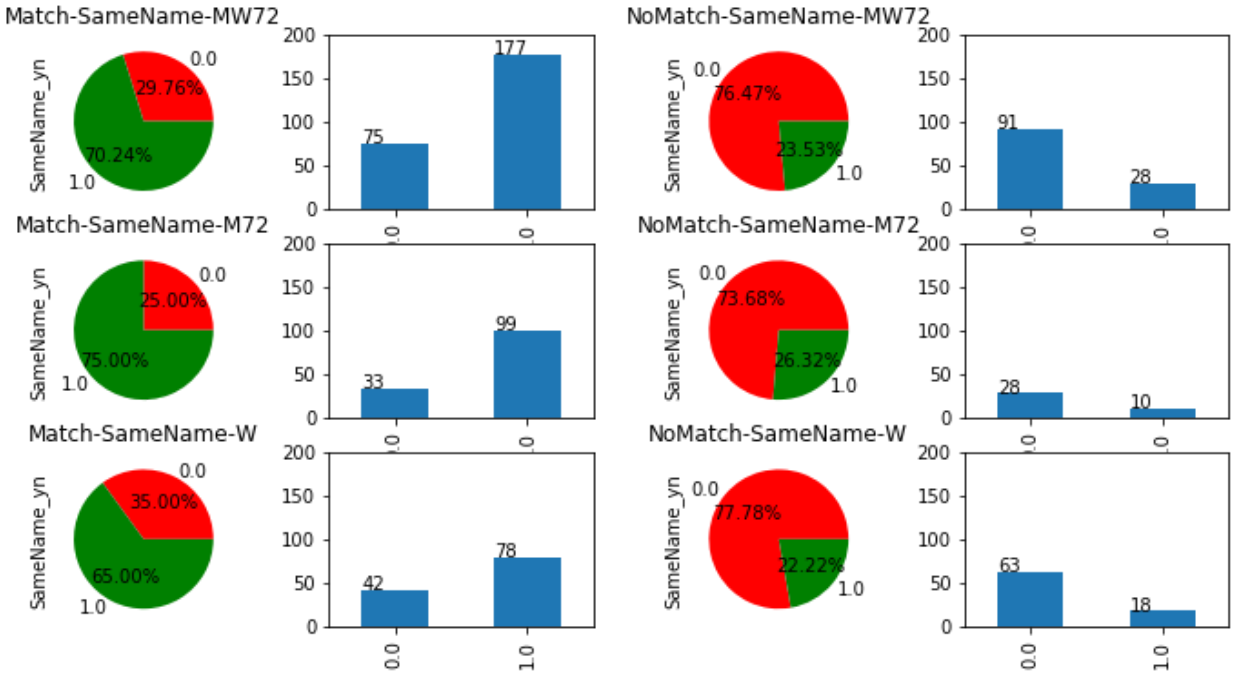
Same Name

The Same Name distribution before and after removing the pre-1972 data shows that most of the excluded data is No Match profiles, and Men have more and a higher percentage of Matches when the name returned with the profile is an exact match with the name sent in the query.



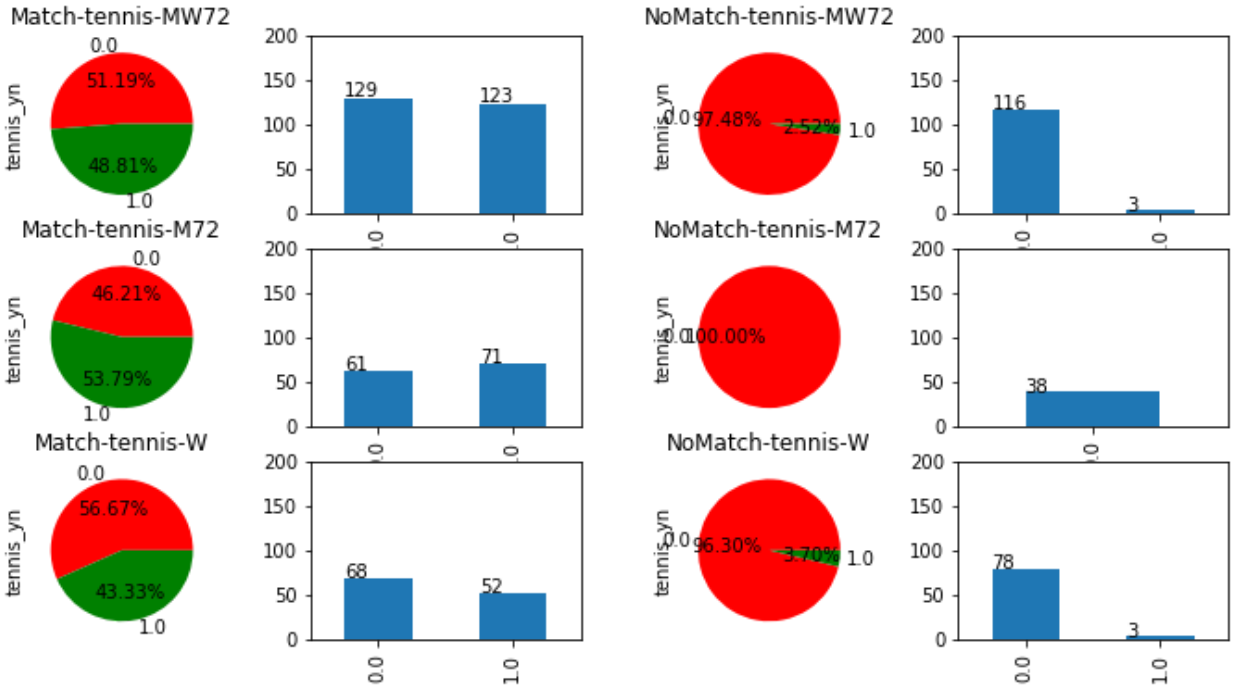
Same Name 1972 or later

After excluding the pre-1972 data, there are significantly fewer NoMatch results where search name is not matching the name returned in the profile returned.



Tennis listed in Activities

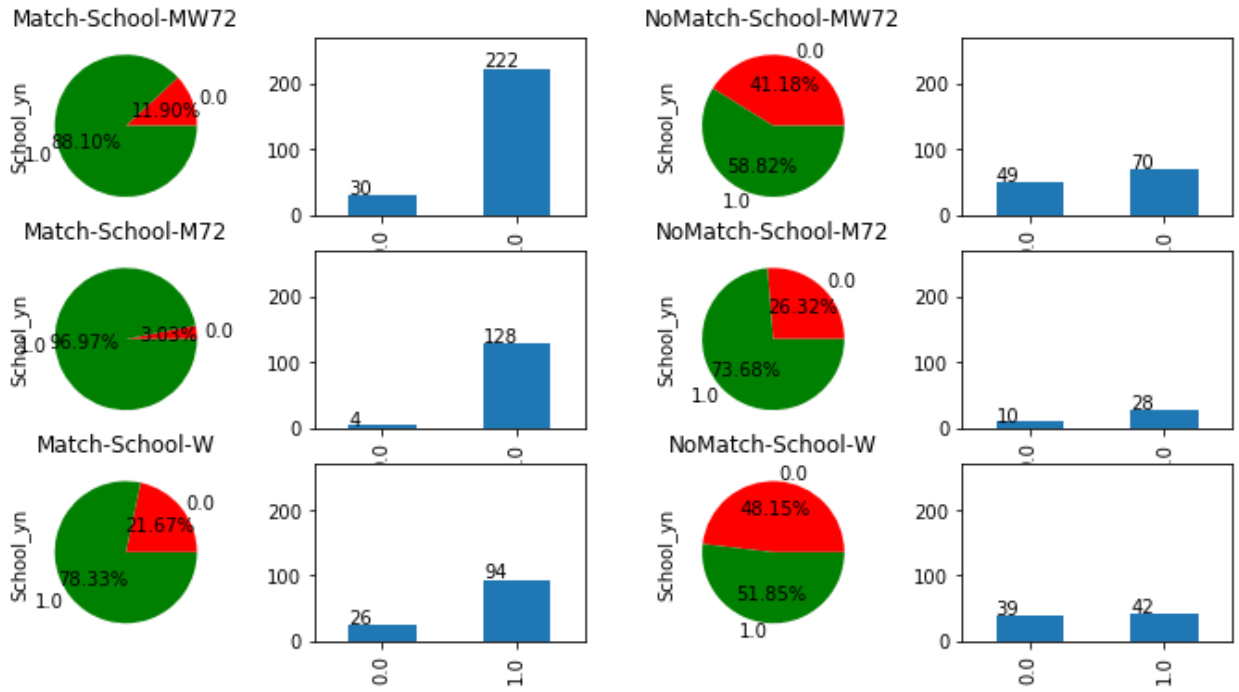
While listing “tennis” in Activities is done roughly half the time on Match profiles, it is almost always missing from NoMatch profiles.



School identified - note a data error in LinkedIn download

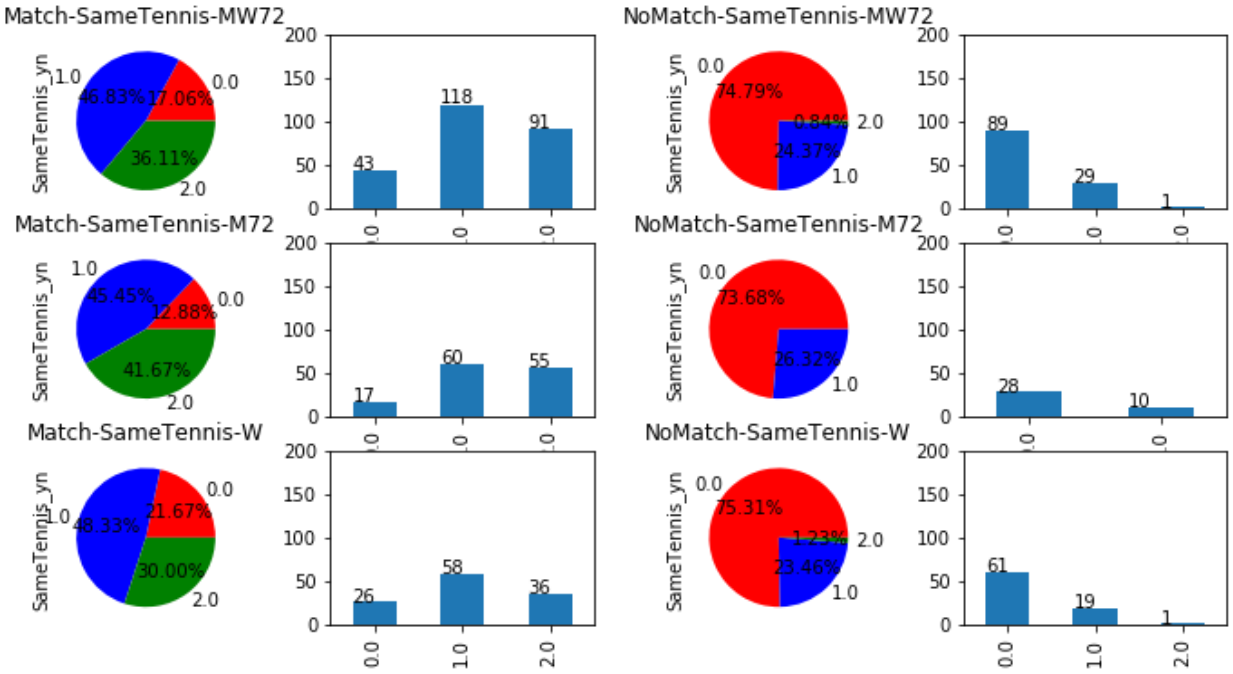
There is a clear pattern that Match profiles identify the school. Even though NoMatch profiles also identify school more often than not, note that school is used in the search criteria so we would expect that even a NoMatch would list the school. There are a few people who are correctly identified but do not list their undergraduate school. This is just another example of why it is not possible to define absolute criteria for the prediction. It appears that Women do not

list the school where they played tennis in both the Match and NoMatch categories, significantly more than Men. Since this does not seem to make sense, the data was reviewed and it appears to be an anomaly in the data collection from LinkedIn. In other words, the download does not return the school in many cases, even though it is listed in the profile. Although we are not using the Company field downloaded from LinkedIn in the analysis, many of these same observations return “Not Found” instead of the company seen in the profile. These xml tags appear to be consistent with those on other profiles when inspecting, so unfortunately it is beyond the scope of this project to figure out why some fields are missing from the LinkedIn downloads.



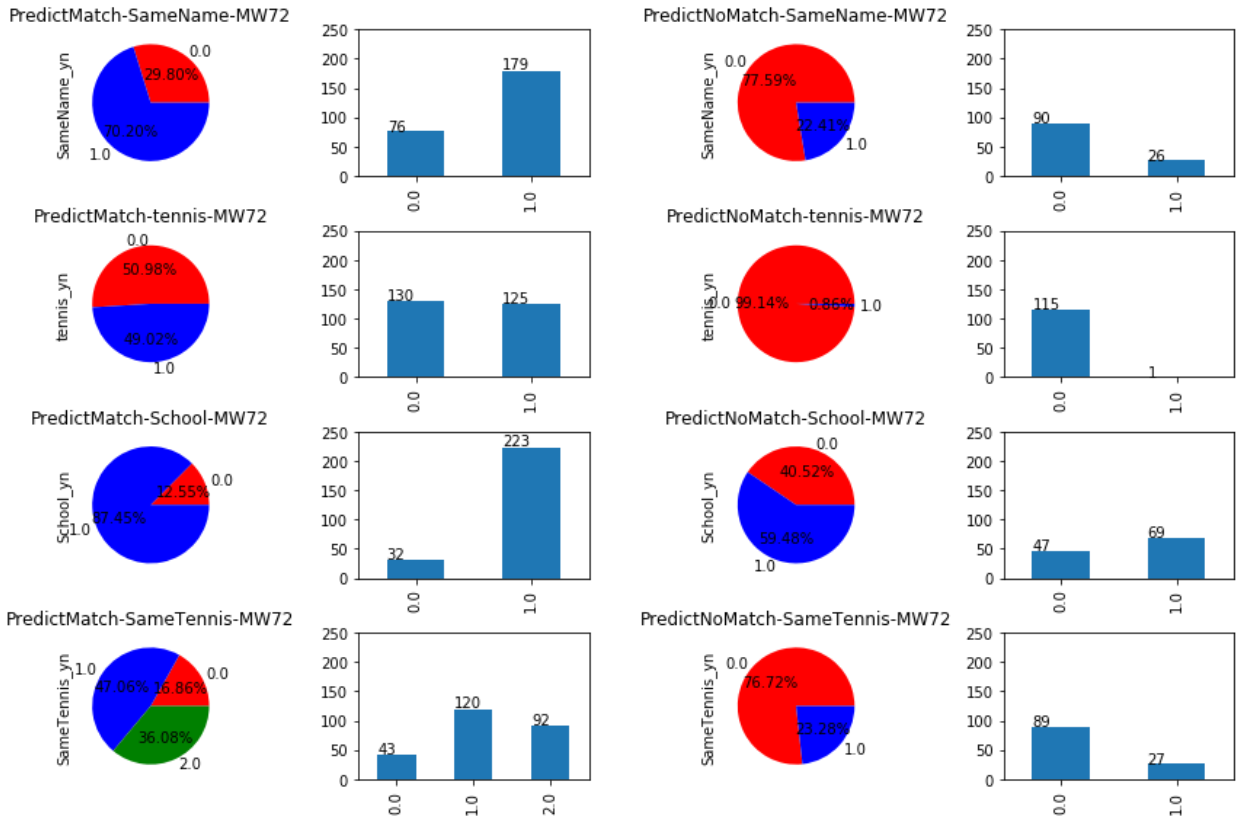
Combining Same Name and Tennis

By adding the 0,1 values for Same Name and Tennis, we can see that it almost never happens that a NoMatch will be positive for both indicators. For Matching profiles, it is worth noting that about 17% are negative for both indicators. This means it is not that uncommon for someone to avoid identifying with these indicators even though they are the person from the team. For that reason, it is very difficult to avoid having at least some wrong predictions in any binary classification model. In other words, because this behavior is not extremely rare, the model will end up having to guess which observations with all the same characteristics must be selected as the counter-intuitive result, in order to achieve a distribution that reflects ratios observed in the labelled results. Given this view, I believe the model performance is close to as good as it could ever be, since the nature of the data does not have a way to completely distinguish a positive result based on features, when different people have the same name.



Prediction Summary - Boolean Features post-72 combined

As the analysis shows there is no significant difference in the gender split for these features, the prediction summary results can be presented without the separation. Note that the predictions are very similar to actual results in the labelled data.



Conclusion

This project summary describes the data for one school, with prediction results that are very accurate. It can sound a little misleading to tout the accuracy of mimicking a result with a true “success” rate of about 45%. Rather than summarize the model performance with statistical measures, it is best to describe how the search results, model predictions, and dataset are useful beyond the numbers. The profile search is highly successful for the younger generation, but returns a lot of wrong profiles for older generations. The model is useful in combination with the full dataset because the user can quickly filter out most of the profiles that are not worth reviewing. At the same time, the full roster list is provided with some information that is useful for performing a regular internet search in order to track down people who are not on LinkedIn. For example, using home town or the school’s town can turn up a person’s former address and possibly link it to a current address. Since the fuzz ratios and boolean are provided, the user can sort or filter out the obvious conditions for detecting false negatives. For example, check all the NoMatch predictions with a number filter on fuzz ratio “>80” to review profiles with a near matching name, since the model needs to guess that some of these are not matching in order to approximate the true frequency. A sample report has been generated for Yale University, using data that was not used at all in training the model. The results have not been labelled, but when performing a sanity check on the fuzz ratios, a new feature was added to the report visualizations. The average of the 5 fuzz ratios was converted to a boolean integer, in order to compare it as a proxy for Same Name, and create the condition being greater than or equal to 80. This helped identify some false negatives while reviewing the data in Excel with filtering. This is the type of sanity check that supplements the model predictions to make them more useful. A separate document with the result summary is provided, as a sample of the deliverable package of a summary report and Excel spreadsheet containing the data.

Future Work

The report is a first draft, and will be improved over time. As the outreach is a collaborative effort, it makes sense to obtain feedback from users on how the fields and data are best explained from their perspective. In addition to a field description summary suitable for a standalone report, instructions on filtering in Excel will be developed. A few more sample reports will be gathered and labelled, and experiments will be run to see whether adding more data and rebuilding the model will improve its performance.