Maya Madhavan

Project Design

Goals

Having just graduated from college and with grad school somewhere on the horizon, I've been thinking a lot about how students apply for schools. When I was thinking about this project, I knew that I wanted to do something related to education. The number of students attending college numbers in the millions and is generally trending upwards. Yet there hasn't been any change in the names of schools that come up in the media, from teachers, and from parents. I personally think that the quality of education depends on both the students and the school, and as such there are many underrated schools out there.

Plan

Given that a student considering colleges probably knows at least one school they want to go to and their chances at that schools, I wanted to suggest to them other schools that were similar but they may not have considered. I planned to do this without including rankings, since they are susceptible to pre-made assumptions. I was also curious to see what schools clustered together to see if there were any better classification systems out there.

Process

Data

Gathering

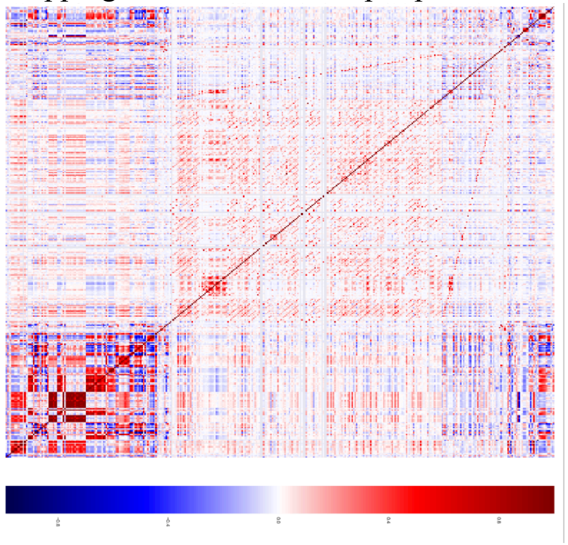
I gathered the data from four sources.

1. College Scorecard: This was my main dataset. It's released every year through the US Department of Education, and is available via csv downloads and and API. I took the CSV download of the information from the 2015-2016 school year, as this was the most year available in this format.
    a. This dataset had 1500+ columns about 7000+ schools across America.
2. US News: Based on my personal experiences, US News rankings are the most commonly used by applicants. This is where I scraped liberal arts college rankings for my initial visualizations, along with tuition and fees, enrollment, location, and whether the school was public or private.
    a. I used some of this in my cleaning
    b. There were ~200 ranked schools
    c. Rankings change only marginally on a year to year basis (see: rankings based on previous assumptions), so there wasn't a major problem with matching the data.
3. Wikipedia: I scraped NCAA divisions from their respective Wikipedia pages. I also collected some more location and school type (public/private) data to use while cleaning.
    a. ~1000 Schools
4. Data.world: There was a premade csv here of US News University Rankings, including the rank and other info that was again useful for cleanings.

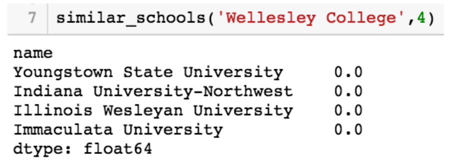
Cleaning

There was a significant amount of cleaning involved with the data. First, I had to get the scraped datasets to look clean, and then I had to impute data. After cleaning the format of individual datasets, I joined the tables in SQL.

I focused only on imputing data for columns that students would be interested In, such as the percentage of white students (diversity), gender, and SAT scores. This was because of the sheer number of nulls and columns in the data. I used the mode/means of given columns based on other ones and filled in with other columns where applicable. I also dropped schools that didn't offer accredited 4 year bachelors degrees because those weren't relevant to my target audience.
EDA
There were enough columns that were highly correlated that I felt comfortable being liberal with dropping features. Here is a pairplot in all its glory for your viewing pleasure.



Algorithms
Content Recommender
I build a cosine similarity based content recommender which gave me results that didn't make sense with what I knew about the colleges I tested it on.



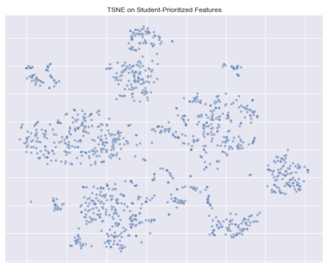
```
similar_schools('Wellesley College',4)

name
Youngstown State University      0.0
Indiana University-Northwest     0.0
Illinois Wesleyan University     0.0
Immaculata University            0.0
dtype: float64
```

Clustering I
This was the major part of my project. I had a class built, which was a pipeline including multiple clustering algorithms and tsnes. It also enabled me to test different PCA components. In terms of clustering, first modified my data so that it was all in int/float format. The remainder of the process was as follows:

1. Dimensionality Reduction: Tested different number of components in
2. Perplexity: Tested different perplexities within TSNE's for visualization
3. Clustering: Attempted the following, modifying parameters on each in order to id the best result (except spectral, due to time constraints): 1. KMeans, 2. DBSCAN, 3. Mean Shift, 4. Spectral Shift
4. Choosing and Analyzing Clusters: Chose the best cluster based on TSNE's, then looked at the closest and furthest points wihin each cluster to discern meaning.
5. Making Recommendations: Created a cosine similarity based function to pull the 3 closest schools to an inputted school within each cluster.

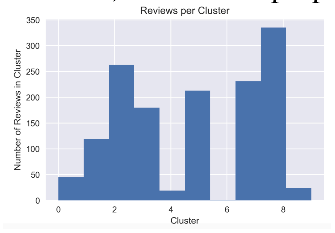
TSNE on Reduced Data

This was a simple visualization of the data with a significantly reduced set of features for each school, for the purposes of future work.



Results

I went with KMeans, with 10 clusters and 100 Components in my PCA, explaining 80% of the variance, and used a perplexity of 30 for my TSNE.



Flask App

The last thing I did was build a flask app whose format was modelled off of the typical application format, wherein students choose 4 schools in the following three categories: reach, target, safety.

<u>Going Forward</u>

Looking Back

I was very pleased with how this project turned out, although I was concerned about the amount of time I'd spent cleaning data. I would have gone ahead and done what's outlined before had I not run into this data roadblock. I would prepare for this ahead of time but spending more time thinking about where I could get the data before starting.

More To Do

As stated in my presentation, I'd like to simplify my model to accommodate students who have no idea what schools they are interested in, i.e. for those who are just starting out with applying (high school juniors). I'd also want to drop acceptance rate, which I'd kept to retain the "reach, target, safety" schema but can be a proxy for rank.

<u>Data: 4 Sources</u>
<u>Tools: SQL, Pandas, Flask, Tableau</u>
<u>Algorithms: Clustering, Recommender Systems</u>