Project Proposal

Westmont College, like any other college or university, receives a large number of applications from prospective students seeking admission every year. What follows is a long process of examining each application and selecting some students to offer acceptance, while rejecting the rest. Some special recruiting occurs, such as with especially appealing students or talented athletes, and the college accepts that while some offers of acceptance made to these students will be accepted, many will be declined. The majority of an incoming first-year class however, is determined through a more general process, for which consultants are hired. These consultants help to determine which applicants to offer acceptance, but more students are accepted than can be accommodated in the incoming class. This is because some of these acceptances will be declined, as with the “special recruiting,” but in the case of the general recruiting this becomes less tolerable. The higher the percentage of those accepting their admission out of those being offered admission, the better the college appears, so the administration would prefer to make less total offers of acceptance, by making less offers to those that will reject them.

The goal of this project is to replace the current "general admission" process with software. This would eliminate the need to hire consultants, thus saving the school a significant amount of money. It is hoped that this software could also more accurately identify students who will accept their offers of acceptance (compared to the current process), so that less offers can be made with a higher rate of acceptance (which is desirable as outlined above). Thus, successfully implementing the project would result in a cheaper, easier, more efficient process for responding to the high volume of applications, and it would produce better results.

The software would be implemented as one of the many varieties of machine learning algorithms. Machine learning algorithms are algorithms which improve their ability to perform some set of tasks (as measured by a specific performance measure) by gaining some form of experience with the tasks. In this case, the task is to determine which applicants from a given pool to offer acceptance, and the performance measure is the percentage of those offered acceptance who confirm. The experience comes from the college’s records of applicants and which of them were accepted, which can be used to “train” the algorithm to produce the desired results. The first step in implementing the software is a brief research phase to determine what form of machine learning algorithm is most appropriate for this particular situation. If the research produces enough confidence in a certain method, this may eliminate the need to implement several different approaches for comparison at a later point. After the initial implementation of the chosen method, more time can be spent improving and fine-tuning the algorithm so that it performs appropriately. Implementation of the algorithm will be the main effort of this project, but it may be necessary to employ some kind of NoSQL (non-relational)

---

database to handle the large amount of data associated with all the applications. The final product should function in way so that it can be given a list of applicants with their relevant data and a desired number of students for the incoming class, and return the same list of applicants separated into those who should be offered acceptance and those who should not be to produce approximately the desired number of confirmed students for the incoming class.

One major potential problem that is generally associated with machine learning algorithms is making sure that the algorithm generalizes well from the data that it was trained on to the “production” data on which it is tested. In this case, this amounts to asking whether acceptance data from years past (the training data) applies to current applicants. Assuming that enough of the factors that influence whether a person confirms their acceptance are included in the applicant data to make a justified decision, and that these factors stay essentially the same from year to year, the algorithm should discover the relevant factors, so this should not be a major concern for this situation. A more pressing concern is the fact that there is no way to know whether students who were not offered acceptance would have confirmed it had they been accepted, which could create a problem for training the algorithm. For instance, if the algorithm suggests offering acceptance to students who were not accepted (using the training data), there is no way to determine whether this is a good or bad decision. One way to solve this issue is to tune the algorithm so that it accepts roughly all the same people who were accepted and confirmed, but rejects all the people who were accepted but chose not to come (for training data). This way the correct number of people are accepted, and in fact essentially the same people end up in the incoming class, but the percentage of people confirming their acceptance is higher.