Will I Get in? - Modeling the Graduate Admission Process for American Universities

Narender Gupta
University of Illinois at Urbana-Champaign
ngupta18@illinois.edu

Aman Sawhney
University of New Mexico
asawhney@unm.edu

Dan Roth
University of Illinois at Urbana-Champaign
danr@illinois.edu

ABSTRACT

We study the graduate admission process in American universities from students' perspective. Our goal is to build a decision support model that provides candidates with pertinent information as well as the ability to assess their choices during the application process. This model is driven by extensive machine learning based analysis of large amounts of historic data available on the web. Our analysis considers factors such as standardized test scores and GPA as well as world knowledge such as university *similarity* and *reputation*. The learning problem is modeled as a binary classification problem with latent variables that account for hidden information, such as multiple graduate programs within the same institution.

An additional contribution of this paper is the collection of a new dataset of more than 25,000 students, with 6 applications per student on average and, hence, amounting to more than 150,000 applications. The dataset covers hundreds of universities over several years, and allows us to develop models that provide insight into student application behavior and university decision patterns. Our experimental study reveals some key factors in the decision process of programs and, consequently, allows us to propose a recommendation algorithm that provides applicants the ability to make an informed decision of programs to apply to given their profile, with high confidence of being accepted.

CCS Concepts

•**Information systems** → **Decision support systems;** *Data mining;* •**Computing methodologies** → **Machine learning;**

Keywords

Graduate Admissions, Decision Support, Learning Model, Latent Variables, Recommendations

1. INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WOODSTOCK '97 El Paso, Texas USA

© 2016 ACM. ISBN 123-4567-24-567/08/06…$15.00

DOI: 10.475/123_4

Every year, thousands of students apply to American graduate programs and in the process, discover that there is a dearth of reliable sources to aid them in making an informed decision. There are several sources that provide admission related statistics, but do not cater to individual needs, thus, leaving the applicant with the only option of guessing and hoping for the best [6, 5]. Even though learning models have been used in variety of real-world applications, there is surprisingly little literature available on understanding admission dynamics and decision making.

A typical university application packet comprises of transcripts, standardized test scores, letters of recommendation, a statement of purpose that expresses student's aims, ambitions and research interests, and descriptive answers to a few additional questions. Test scores include GRE[1], language test scores - such as TOEFL[2] or IELTS[3] etc. Universities, then, evaluate application packets based on rules or heuristics which are unknown to students, and release decisions. Since requirements, deadlines and the specific process to meet them is university specific, the applicant needs to first choose the universities he would apply to.

Given the uncertainty, a naive solution is to apply to a large number of universities. But, the more the number of applications, the higher the investment of time and energy. This also implies a large monetary investment, which is a major concern for applicants from developing countries. One of the strategies to circumvent this is to categorize the universities into buckets so that one only applies to a few representatives from each category. A popular scheme includes three categories: *Dream*: where the chances of admission are slim; *Reach*: where the chances of admission are decent; *Safety*: where there is a fair certainty of being accepted [18]. Multiple admission offers resulting from these decisions allow the applicant to choose, suboptimally, their best option. The description of these categories is very subjective, and even more subjective is the applicant's ability to predict his probability of *Admit* i.e. chances of being admitted to a given program [18]. This prediction is generally based on hearsay or semi-informed opinions, resulting in confusion and a waste of resources for both the applicant as well as the university.

In this paper, we address this problem by developing a machine learning approach which enables applicants to make informed decisions by evaluating their chances of admission. We suggest a latent variable based generative modeling ap-

[1]Graduate Record Examination www.ets.org/gre
[2]Test Of English as Foreign Language www.ets.org/toefl
[3]International English Language Testing System www.ielts.org

proach which outperforms existing techniques significantly. We analyze our system from students' perspective but it can be easily extended to university's perspective as well.

Some researchers have briefly reflected on the process of decision making, but only qualitatively [23].The work done by Waters et al. models the problem from the university's perspective [25]. They used a learning approach to aid the university admission committee by identifying the candidates that are unlikely to be offered admission. Their model is quite simplistic, considering the problem as a straightforward classification problem (via Logistic Regression) without attempting to reveal the diverse and rich patterns in the data. More importantly, their approach is university centric and does not provide any support to the decision process of applicants. The primary reason for the lack of such endeavours is the unavailability of a relevant dataset. One contribution of our work is the creation of such a dataset, which we will make available to the community. The dataset allowed us not only to determine the acceptability of an application but also suggest better choices.

Works such as Bruggink et al. and Moore et al. utilize domain knowledge to build statistical models[10, 19]. Bruggink et al. model undergraduate university admissions to a private liberal arts college. The model treats application components as independent variables and assumes the decision to be dependent (*Admit* or *Reject*) on these. The independent variables include GPA [4], SAT scores, other academic scores and extracurricular factors all of which have been quantized. Beyond strong statistical assumption, this approach assumes that the modelled university (and its application pool) provide a good representation of the whole distribution. Our study shows that this is not the case because different universities focus on different features, and hence produce decisions differently. We conclude that learning a decision model should be done separately for each university, if possible. Moore et al. model the problem with rule induction using ID3 algorithm. Such an approach without care for bounded depth is prone to overfitting. Similar to Bruggink et al. the model is centered around one university and considers very small applicant sample size.

2. PROBLEM MODELING

2.1 Dataset

There are several online resources where applicants share their admission experiences [17, 4]. We scraped the data present on Edulix [17]. It is an active resource which hosts applicant profiles from all over the world. GRE scores, undergraduate university name, GPA, TOEFL scores and other accomplishments such as work experience and research publications pertinent to the graduate admissions are reported in the profile. In addition, users mention the universities that they applied to and the result of each application (*Admit*, *Reject* or *Result Not Available*). Since the data is self reported, it had some erroneous records. We fixed these problematic entries using techniques explained in Section 3.2. In this paper we focus on modeling admission to computer science graduate programs, which form the plurality of our data. Our experiments are conducted only on this subset of the data. A few of the features of the resulting dataset for the computer science applications are in Table 1.

[4]Grade Points Average

Table 1: Features of Data

General Features	
Total number of users before sanitization	36,207
Total number of users after sanitization	26,148
Features for CS related dataset	
Number of users	10,788
Application year range	[2001 2015]
Median Application Year	2013
Most Frequent Application Term	Fall
Number of universities with reported data	313
Number of applications per student (Mean)	6
Number of applications per university (Mean)	51
Number of undergraduate universities	2353
Degrees sought	[MS, PhD]

2.2 Supervised Learning

Each university offers binary decision to applicant (*Admit* or *Reject*) and this decision is independent of the decision of other universities. Hence, the overall problem can be modeled as a set of individual binary classification problems. Supervised learning algorithms can be trained using features extracted from a labeled dataset and evaluated for performance.

Features: The dataset contains several fields such as standardized test scores and academic history records. We extract several numerical features from these such as GRE test scores (AWA[5], Verbal & Quantitative), undergraduate GPA, language test scores (TOEFL), as well as categorical features such as program applied to (e.g. MS, PhD), term (e.g. Fall, Spring) etc. These features are used for learning Logistic Regression, Support Vector Machine and Random Forest. Since most work available in the literature is confined to approaches we have mentioned so far, we'll regard it as the baseline for any novelties that we propose.

Ensemble Learning: To improve the system performance, we use ensemble learning. Training decision trees without bounded depth is prone to overfitting [22, 20]. But we can hope to generalize better if we use several limited-depth decision trees using partial data, and then feeding each decision as a feature into another regularized classifier. We create d such limited-depth decision tree classifiers where each classifier is trained on a bootstrapped sample from the original dataset [13]. Bootstrapping allows us to have a tunable number of approximate feature representations instead of low number of exact features. Constrained by the variance-bias trade-off, the second classifier captures variance of data through multiple underlying decision trees while keeping a limit on its own variance by choosing simple models such as linear separators e.g. soft-margin support vector machine. Corresponding to decision of each such classifier, we get a feature for the next classifier.

2.3 Latent Variable Based Approach

For a few universities, we noticed that our discriminative classifiers do not perform as expected. We attribute it to the fact that some universities offer multiple degree programs that might target different kinds of applicants and have different admission criteria, e.g. professional master's

[5]Analytical Writing Analysis

versus thesis master's program at UIUC[6]. Certain other universities, such as CMU[7], offer specific programs at same degree level such as master's degree in Machine Learning versus HCI[8] which fall under the purview of Computer Science. These distinctions, however, are not captured in the dataset, and are thus *hidden* from our models. This distribution causes difficulties for linear separators which assume that data is coming from single source. In case the data is coming from multiple sources, it is impossible to linearly separate the data using a single classifier. To accommodate these phenomena, we use an *Expectation-Maximization (EM)* algorithm that allows us to learn a model with a latent variable, capturing the missing information. The rest of this section, describes how we model the problem in this case.

EM is one of the very effective techniques for finding a *Maximum Likelihood Estimate* with *hidden variables* [12]. For the sake of brevity, we are not going into details of *EM*. We will be using a setup similar to the one Grove et al. used in [15].

We assume a latent (hidden) variable called Program Type, z, which might carry one of multiple values. These values can hold different semantics based on university, e.g. different for UIUC and CMU, and, hence, we refer to them simply by the value assigned by the model to the hidden variable such as *1, 2 ... k* etc. Once we learn the most likely model with the hidden variable z taking k values we can use it in two different ways. In a soft-boundary setting, each student can be from either Program Type with some probability. In a hard boundary setting, the most likely cluster (program) completely owns the student record, and we can learn individual linear classifiers for each cluster. A student belongs to cluster $z = i$ if:

$$P(student|z = i) > P(student|z = j), \forall j \neq i \tag{1}$$

where

$$\sum_j P(student|z = j) = 1 \tag{2}$$

We assume that an applicant *belongs* to a program $z \in \{1, 2, ..., k\}$ with probability $\alpha_r = p(z = r)$. Given this program, feature x_i of the applicant x is generated independently by a Gaussian distribution with model parameters (μ_i, σ_i). Hence, we aim to split Gaussian mixture of data into individual models using *EM* setting.

Let us define a hidden variable, $z \in \{1, 2, ..., k\}$. A student record (sample), x, consists of *(n+1)* features,

The likelihood of sample is given by

$$P(x) = \sum_z p(x|z)p(z) \tag{3}$$

Incorporating each feature probability, the likelihood of the data sample can be expressed as:

$$P(x) = \sum_z \left(p(z) \prod_i p(x_i|z) \right) \tag{4}$$

Starting with an initial set of parameters θ, the probability that a data point $x^j = x^j = (x_0^j, x_1^j, ...x_n^j)$ comes from each of the k values of z is given by:

$$P_r^z = p(z = r|x^j) = \frac{p(z = r) \prod_i p(x_i^j|z = r)}{\sum_z \left(p(z) \prod_i p(x_i^j|z = r) \right)} \tag{5}$$

Let $p_i^z = p(x_i|z = r)$, then we can compute the expected log-likelihood as follows:

$$\begin{aligned} E(LL) &\equiv E\left(\sum_j \log P(x^j|\alpha_z, p_i^z) \right) \\ &\equiv \sum_j E\left(\log P(x^j|\alpha_z, p_i^z) \right) \\ &\equiv \sum_j \left(\sum_z P_z^j . \log P(z, x^j|\alpha_z, p_i^z) \right) \\ &\equiv \sum_j \left(\sum_z P_z^j . \log(\alpha_z . \prod_i p(x_i^j|z)) \right) \end{aligned} \tag{6}$$

Assuming numerical features to be generated from a Gaussian distribution with parameters (μ, σ), above equation can be expanded as:

$$\begin{aligned} E &\equiv \sum_j \left(\sum_z P_z^j . \log \left(\alpha_z . \prod_i \frac{1}{\sigma_i^z \sqrt{2\pi}} \exp\left(-\frac{(x_i^j - \mu_i^z)^2}{2(\sigma_i^z)^2} \right) \right) \right) \\ &\equiv \sum_j \left(\sum_z P_z^j . \left[\log \alpha_z + \sum_i \left(-\log \sigma_i^z - \frac{(x_i^j - \mu_i^j)^2}{2(\sigma_i^z)^2} \right) \right] \right) \end{aligned} \tag{7}$$

Differentiating with respect to all parameters, we can find new values of α_z, σ_i^z, and μ_i^z, for which the expected (log) likelihood receives an extremal value. Since, α can only assume $k-1$ independent values because of k states of z, we have:

$$\alpha_k = 1 - \sum_{i \in [1...k-1]} \alpha_i \tag{8}$$

Using this equation and differentiating Eq (7) partially with respect to model parameters, we get following update rules:

$$\alpha_z = \begin{cases} \alpha_k \frac{\sum_j P_z^j}{\sum_j P_k^j} & \text{if } z \neq k \\ 1 - \sum_{i \in [1...k-1]} \alpha_i & \text{if } z = k \end{cases} \tag{9}$$

$$(\sigma_i^z)^2 = \frac{\sum_j \sum_z P_z^j (x_i^j - \mu_i^j)^2}{\sum_j \sum_z P_z^j} \tag{10}$$

$$\mu_i^z = \frac{\sum_j \sum_z P_z^j x_i^j}{\sum_j \sum_z P_z^j} \tag{11}$$

Using above update rules, and various initializations for the latent variables, we performed multiple experiments with EM. Current results are reported for z=2 in a hard-boundary setting.

3. EXPERIMENTAL EVALUATION

Our experimental study is designed to investigate the following issues:

- The ability to make reliable prediction on sparser label for a university for any student.

[6]University of Illinois at Urbana-Champaign
[7]Carnegie Mellon University
[8]Human Computer Interaction

- The ability to make reliable prediction on whether a specific student can be admitted to a given program.
- Our ability to identify sub-programs in a given university, and its significance on the performance of our admission model.
- Understanding the contribution of factors to admission.
- Understanding the differences among universities in terms of their admission decisions.

3.1 Evaluation Metric

Before we evaluate performance, it is important to understand the appropriate performance metric for this task. Prediction accuracy is a biased metric in this case, because most universities have moderate to heavy imbalance in terms of the presence of labels (*Admit, Reject*) and a high accuracy doesn't necessarily indicate effective learning. Some of the universities have as low *Acceptance Ratio* as 8% while others accept more than half of the candidates that apply.

$$Acceptance\ Ratio = \frac{count(Admit)}{count(Admit)\ +\ count(Reject)}$$

In such an imbalanced label distribution, simple baseline of dense label assignment as prediction will result in high accuracy, without the need to learn anything. Hence, we choose F1 as our evaluation metric which takes into account not just the correct number of predictions made for the label (*Precision*), but also the ratio of predicted true labels out of total true labels (*Recall*). A natural choice for the label is *Admit*, since it is what any applicant cares about. But there are cases when it makes the problem suspiciously easier from theoretic standpoint. Consider two simple cases with following label distributions in data, and corresponding simple baselines where we blindly assign *Admit* prediction to each of examples:

1. $Admit : Reject = 10 : 90$
   $Precision = \frac{1}{10}$, $Recall = 1$, $\mathbf{F1(Admit)} = \frac{\mathbf{2}}{\mathbf{11}} = \mathbf{0.18}$
2. $Admit : Reject = 90 : 10$
   $Precision = \frac{9}{10}$, $Recall = 1$, $\mathbf{F1(Admit)} = \frac{\mathbf{18}}{\mathbf{19}} = \mathbf{0.95}$

It is evident that valuating F1 on the sparse (less frequent) label provides stricter bounds on the performance. Hence, from a machine learning perspective, it is easy and uninteresting to predict for the denser label, but from a student's perspective, it makes sense to predict *Admit*. Hence, we evaluate our models on both of the metrics i.e. F1 over sparser label ($F1_{sparse}$), as well as F1 over *Admit* ($F1_{admit}$).

Let $U = \{1, 2, ..., N\}$ is the set of N universities. Using Table 2, $F1^{(i)}$ for the university i is defined as:

$$F1^{(i)} = \frac{2 \times Precision^{(i)} \times Recall^{(i)}}{Precision^{(i)} + Recall^{(i)}}$$

$$Precision^{(i)} = \frac{TP}{TP + FP}$$

$$Recall^{(i)} = \frac{TP}{TP + FN}$$

$F1$ over the set U is defined as the statistical mean of individual $F1$s i.e.

$$F1_{sparse} = \langle \{F1^{(i)}_{sparse} \forall\ i \in U\} \rangle$$

Table 2: Classification Context

		True Condition	
		Admit	Reject
Predicted Condition	Admit	TP	FP
	Reject	FN	TN

$$F1_{admit} = \langle \{F1^{(i)}_{admit} \forall\ i \in U\} \rangle$$

The rest of this section describes the details of our experimental study, and its results. In all our experiments we are reporting average F1 over 5-fold cross-validation.

3.2 Data Cleaning

Since the data is self reported, it had some erroneous records. We fixed these problematic entries using following techniques:

- *Missing Values*: If the record did not have mandatory fields such as GRE Verbal & Quantitative, or GPA, then such a record was completely deleted. Missing year and missing term were replaced with the mean of existing values.
- *Out of Range Errors*: Records which had invalid values such as negative GPA, or out of range value for GRE score components were removed in a rule-based fashion.
- *Illegal Data-types*: Records having incompatible datatypes such as string for GPA, or numerical value for degree were removed.
- *Quantitative Normalization*: Undergraduate institutions all over the world follow different scales for reporting GPA. Similarly, GRE and TOEFL tests have undergone various scale transformations over the years. As a standardization measure, we mapped these fields linearly to a scale of [0,100].
- *Categorical Normalization*: Sometimes students refer to the same degree using various acronyms such as *BE* or *BEng* for *Bachelor of Engineering*. Similar is the case for departmental majors such as *CS* or *Comp Science* for *Computer Science*. Also, sometimes there are spelling mistakes and typos in these fields. Each of these fields is, hence, normalized to manually created codes e.g. {*ba,bs,be,btech etc.*} instead of original strings. The code assignment was done using regular expressions or a manual mapping of string value to code.
- *University Normalization*: An undergraduate university might be referred to by the differing names due to reasons such as usage of a popular acronym or spelling errors. We mitigated this problem by mapping the university names to their unique website homepage using web search engine. Under this scheme, a program queried a search engine which returned a ranked list of relevant URLs. These URLs were then filtered based on heuristics (e.g. remove blog or social network URLs, prefer *.edu* domains and so on) and most relevant URL was assigned to the university. These assignments were then manually verified for consistency.

- *Labels*: There were no such labels present in dataset as *Waitlisted*, or *Conditionally Accepted*. We excluded any application that was classified as *Result Not Available* because it simply represents either a missing value or a pending decision.

3.3 Discriminative Classifiers

We ran multiple experiments with various classifiers such as - SVM with linear kernel or Radial Basis Function kernel [11], [24], Logistic Regression, Adaboost [14] with decision trees [9], and Random Forest [8]. We also experimented with all of the above classifiers with balanced class weights by adjusting them inversely proportional to the class frequency in input data. Each of these setups is used in two different settings:

- **Simple features**: Features extracted from student records
- **Tree features** Training d decision tree classifiers with bounded-depth and then using predictions of these classifiers as features. Each decision tree is trained on uniformly sampled 50% data restricted to maximum depth of 3.

Using grid search in the range [10,1000], we found that d=60 yields maximum average F1 over all universities. These classifiers can provide the probability of the label as well which we utilize in Section 4. These experiments were conducted using Scikit-learn [21]. Table 3 lists the tuned parameter values for different classifiers.

Table 3: Classifier parameters tuned by grid search

Classifier (Param)	Range	Best Value
SVM(Linear): C	[1,100]	29.55
$SVM_{balanced}$(Linear): C	[1,100]	26.29
SVM(RBF): C	[1,100]	3.68
SVM(RBF): γ	[0.01,1.0]	0.05
$SVM_{balanced}$(RBF): C	[1,100]	2.59
$SVM_{balanced}$(RBF): γ	[0.01,1.0]	0.11
Logistic: C	[1,100]	11.29
Logistic: penalty	L1,L2	L1
$Logistic_{balanced}$: C	[1,100]	14.91
$Logistic_{balanced}$: penalty	L1,L2	L1
Adaboost: No. of estimators	[1,100]	68
Random Forest: depth	[1,5]	3
Random Forest: depth	[1,5]	3

Table 4 lists $F1_{sparse}$ and $F1_{admit}$ for both the schemes i.e. Simple features as well as Tree features. It should be noted that Logistic regression as used by Water et al [25] does provide the best results if features extracted from the application are fed into the classifier directly, but these decisions can be improved significantly by other forms of modeling such as decision stumps, and even further by using latent variable based generative modeling as detailed in Section 3.5.

Figure 1 plots $F1_{sparse}$ as a function of the presence of sparser label. As expected, performance of the model gets better as the sparse label ratio gets closer to 0.5 (i.e. sparse label percentage gets closer to 50%). It is also evident from the figure that all the individual $F1^{(i)}$s for decision tree featured model are performing better than simple feature model. The curves, fitted to the individual points by using moving average over a window of 5, show similar trend.

Similar is the case for $F1_{admit}$ when it is plotted against *Acceptance Ratio*, in Figure 2.

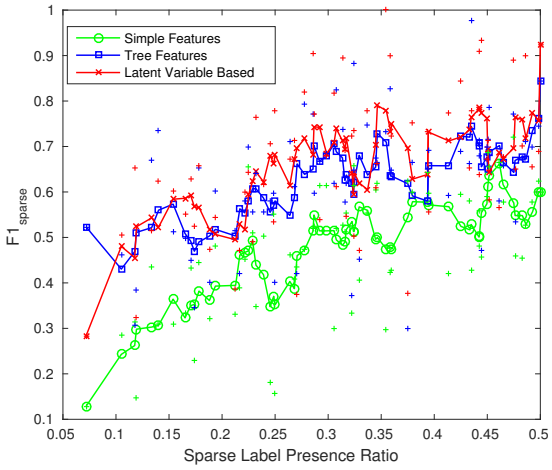

Figure 1: $F1_{sparse}$ as a function of presence of sparse label. The curve is fitted with moving average function of window size 5.

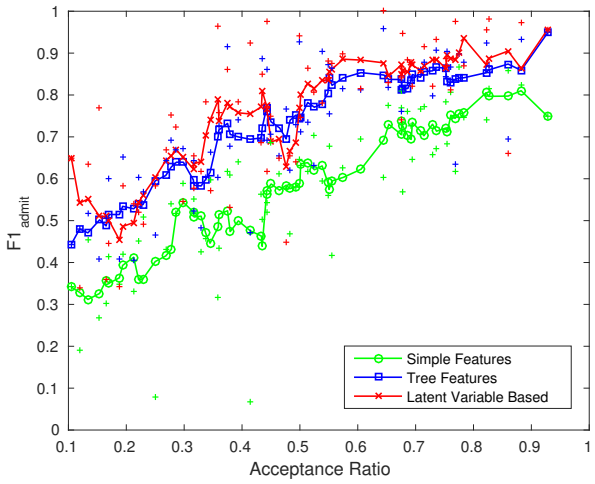

Figure 2: $F1_{admit}$ as a function of Acceptance Ratio. The curve is fitted with moving average function of window size 5.

Table 4 shows that $F1_{admit} > F1_{sparse}$ for all the schemes. This is because *Acceptance Ratio* for some universities is greater than 0.5. Predicting *Admit* in this case results in higher $F1$. Similar to the axis of Figure 1 and Figure 2 can be other sorting criteria which can reveal interesting patterns. One such criteria is a proxy of the *reputation* of the university i.e. University ranking. We chose this to be the US News Graduate School Ranking [6], primarily, because of its popularity among applicants. Whenever a rank was not available in this resource, a similar resource was consulted [5, 1, 3]. In US News ranks, sometimes multiple adjacently ranked universities were stacked up on a single rank, and the resulting emptied out slots were left vacant. We flattened

Table 4: F1 over different classifiers and schemes

Classifier / **Scheme**	**Simple Features**		**Tree Features**		**Latent Variable Based**	
	$F1_{sparse}$	$F1_{admit}$	$F1_{sparse}$	$F1_{admit}$	$F1_{sparse}$	$F1_{admit}$
SVM (Linear Kernel)	0.34	0.52	0.60	**0.73**	0.66	0.75
$SVM_{balanced}$ (Linear Kernel)	0.46	**0.58**	0.61	0.70	0.66	0.75
SVM (RBF Kernel)	0.18	0.46	0.57	0.70	0.60	0.76
$SVM_{balanced}$ (RBF Kernel)	0.09	0.41	0.35	0.57	0.37	0.62
Logistic	0.36	0.56	0.57	0.70	0.60	0.72
$Logistic_{balanced}$	**0.48**	0.57	0.57	0.68	0.62	0.72
AdaBoost	0.37	0.54	0.55	0.70	0.59	0.71
Random Forest	0.24	0.49	0.59	0.71	0.62	0.74
$Random\ Forest_{balanced}$	0.42	0.56	**0.62**	0.72	**0.65**	**0.77**

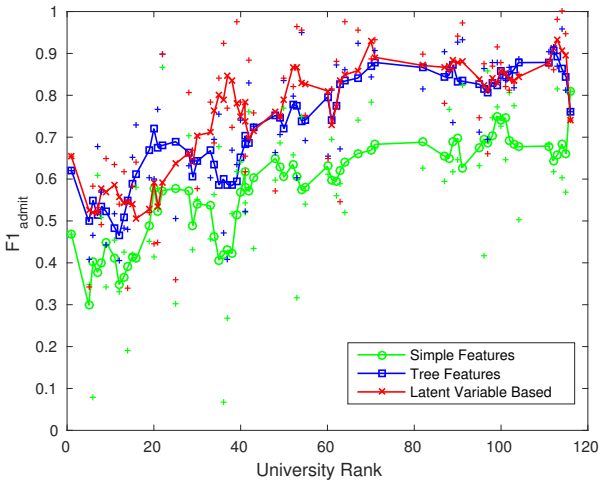

Figure 3: $F1_{admit}$ as a function of Graduate University US News Rank. The curve is fitted with moving average function of window size 5.

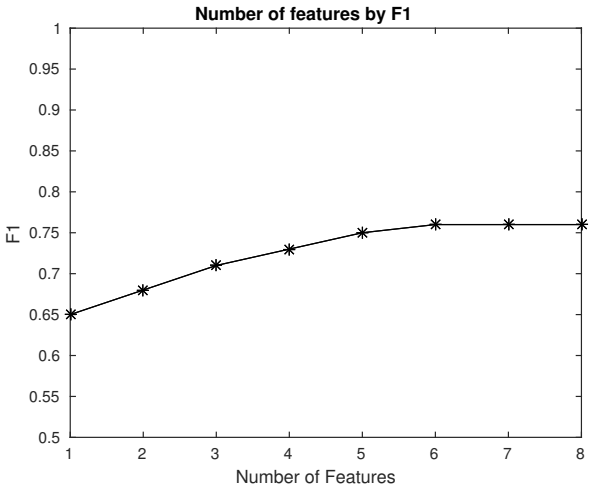

Figure 4: $F1_{admit}$ as we keep on increasing features on top of GPA. Refer to Table 5 for feature corresponding to index number on X-axis.

each stack to its nearest available slots. A high rank means high numerical value of the university rank, which means lower reputation for the university, and vice versa. Thus, a university which is regarded as the best will have the lowest possible rank under this scheme. Interestingly, this ranking scheme has a high positive value of Pearson's correlation coefficient with *Acceptance Ratio*.

$$\rho(Acceptance\ Ratio,\ University\ Rank) = 0.65$$

Such a high correlation suggests a similar trend for $F1_{admit}$ for University ranking, which is confirmed in Figure 3. Figure 3 also suggests that it gets easier to predict *Admit* as we go down the rankings. Despite disagreeing on absolute values, both $F1_{sparse}$ and $F1_{admit}$ show similar trends, and given the student's perspective as explained earlier, we'll be using $F1_{admit}$ for the rest of our experiments as well as recommendation system.

3.4 Feature Selection

These experiments were aimed at understanding the value in each feature. We trained multiple classifiers using single features and evaluated their performances. Then we iteratively added more features to each of the classifiers and evaluated gain in performance. Each feature, when considered individually, is the only classifying parameter. We call its corresponding $F1_{admit}$ result the Discriminative Power of feature. Although the addition of features one-by-one can theoretically lead to combinatorial explosion, the limited number of original features available in our case prevents this from happening. It was observed that undergraduate GPA has the highest discriminative power and has an average $F1_{admit}$ over all universities close to 0.65. Figure 4 shows how overall $F1_{admit}$ increases if we sort the features into descending order of discriminative power and keep on adding them to the classifiers.

We observed from experiments that GPA results in a higher $F1_{admit}$ score compared to any other single feature for any university. Also, if we calculate $F1_{admit}$ by excluding individual features, exclusion of GPA causes maximum loss in $F1_{admit}$. Both of these observations lead to the conclusion that GPA has the highest discriminative power among all available features. The result is intuitive as it validates the expectation that, broadly speaking, GPA is the prime factor in the admission process. It can also be seen that as the number of features that are considered is increased, performance goes up and is the highest when all the features are considered. Table 5 lists individual discriminative powers of each feature, as well as cumulative power for i features i.e.

Table 5: Discriminative Power of each feature

Index	Feature Name	$F1_{admit}$	Individual F1
1	GPA	0.65	0.65
2	GRE Quant	0.68	0.53
3	GRE Verbal	0.71	0.58
4	GRE AWA	0.73	0.45
5	TOEFL	0.75	0.58
6	Program	0.76	0.31
7	Term	0.76	0.35
8	Previous Department	0.76	0.42

Table 6: F1 without each feature. Less F1 due to missing feature indicates more discriminative power of that feature.

Ignored Feature Name	F1
GPA	0.7234
GRE Quant	0.7415
GRE Verbal	0.7422
GRE AWA	0.7491
TOEFL	0.7455
Program	0.7654
Term	0.7647
Previous Department	0.7518

when features $1, ..., i$ are used for classification. Table 6 lists loss in $F1_{admit}$ due to exclusion of each feature during classification. Fig 4 plots the cumulative discriminative power when we keep on adding features.

3.5 Latent Variable Based Approach

The EM model formulated in Section 2.3 was used to cluster students into different groups, representing potential programs. Subsequently, individual classifiers were learned for each of the clusters. Results for few of the universities for which significant growth was observed are listed in Table 7. As per the model assumption, EM bifurcates the data into two clusters (z=2), each of which can be separated in a better way than the earlier cumulative cluster thereby increasing the performance of the models significantly. Improvement in $F1_{admit}$ due to EM clustering is reported in Figure 1 and Figure 2.

Our model before the use of EM relied on the fact that data for each university is coming from a single source. The improvements in $F1_{admit}$ as result of splitting the data according to EM formulation indicates that our model is able to capture underlying different distributions of source data. Also, since we know that UIUC offers different degree programs (Professional, Thesis), and CMU offers different specifications (Machine Learning, HCI etc) for the data reported as CS, it is probable that several other universities have more than one underlying distribution because of other factors. Fig 3 shows overall increase in $F1_{admit}$ over all of the universities using EM splitting in two clusters (z=2). Table 7 reports some of the universities where EM modeling caused roughly 10% or more relative improvement. Although the value of z is also a tunable parameter of the model, we experimented only with $z = 2$. As z, and correspondingly the number of clusters, increases we expect sparser clusters and hence prone to overfitting.

Table 7: Gain in F1 score due to EM clustering

University	Tree	EM + Tree	EM Gain
UCSC[9]	0.58	0.84	0.26
SJSU[10]	0.62	0.86	0.24
UCLA[11]	0.45	0.68	0.23
UMD[12]	0.43	0.66	0.22
SUNY Binghamton[13]	0.76	0.94	0.17
UT Austin[14]	0.57	0.74	0.17
UC Boulder[15]	0.80	0.94	0.14
TAMU[16]	0.75	0.86	0.11
UIUC	0.57	0.65	0.08
CMU	0.66	0.71	0.05

3.6 Understanding Institution Rankings

There is evidence in literature that if an applicant belongs to a *reputed* institution it is regarded as grounds for, at least, some liberality in the way the admission decision would swing [23]. We aimed to qualify this notion by formally validating the assumption - *Undergrad university ranking plays an important role in admissions.* Proving or disproving such an assumption required two-fold experiments:

1. Does university ranking play any role in admission?

2. If yes, what is this rank list?

First, we investigated if the notion of an undergraduate institution‘s rank or category even exists. If it does, providing this extra knowledge should help improve the classifier‘s performance. Our dataset has applicant records from thousands of undergraduate institutions across many countries. The largest resource for university rankings we were able to find was *Ranking Web of Universities (webometrics)* which ranks 11,701 universities across more than 150 countries and territories [3]. Even such a large resource was not exhaustive enough and did not have any ranking information for undergraduate universities of more than 45% applicants. Hence, apart from feeding *webometrics* ranking to the classifier as feature, we set up several instances of this experiment by using other ranking proxy signals. These proxies included rankings provided by US News [6], QS (Top Universities) rankings [5], Shanghai rankings [1] and other lists provided by various private or government agencies such as 'List of Institute of National Importance in India' [2]. Since extending the large ranking list of webometrics at such its original finegrain scale was impractical, we divided universities into four categories, as follows:

[9]University of California Santa Cruz
[10]San Jose State University
[11]University of California Los Angeles
[12]University of Maryland College Park
[13]State University of New York Binghamton
[14]University of Texas Austin
[15]University of Colorado Boulder
[16]Texas A and M University College Station

- *Rank-A*: Institutions ranked as top tier and widely recognized.
- *Rank-B*: Institutions ranked in the middle tier or recognized regionally.
- *Rank-C*: Institutions ranked in the low tier.
- *Rank-D*: Institutions that are neither recognized nor ranked.

Our hypothesis was that reducing the finegrain rankings to broad categories could reduce variance, but such a scheme could easily assign a category value to each university, thus removing the missing value problem for ranking. We expected that if a largely agreed upon rank-list existed, and if our proxies were representative of such a list, then this rank-list should be able to provide gain to the classifier. At the same time, any other list which deviates drastically from such a list should not provide comparable gain during classification.

This category distribution was referred to as **'Original Rank List' (ORL)**. ORL had following category distribution: Rank-A=47, Rank-B=217, Rank-C=363, Rank-D=2354. Next, we **consciously shuffled** this list using following rules:

1. A university can have either the same category as it originally had, or it can move to its closest category, e.g. Rank-B can move to either Rank-A or Rank-C. The probability of an institution moving to neighbor category is linearly proportional to the target size.
2. Each category still has the same number of institutions as it originally had.

Since, we shuffled the institution categories based on precise rules, we called the result as **'Consciously Shuffled Rank List' (CSRL)**. We were taking into account that ranking of a university varies with ranking agencies or regions. In addition, we maintained the original category distribution (size of category) of the institutions. **'Randomly Shuffled Same Distribution Rank List' (RSSDRL)** was created by assigning a randomly chosen category to each institution but by maintaining original category distribution. Finally, we created a **'Randomly Shuffled Uniform Distribution Rank List' (RSUDRL)** by assigning a random category to each institution, without the constraint of maintaining original distribution. In RSUDRL, each category has uniform probability of occurring within the rank list. These rank-lists will also be released as part of the dataset.

Figure 5 indicates that the addition of rank of the undergraduate institution feature led to the gain in performance. We evaluated the gain in terms of statistical significance over 100 iterations and it was significant with $p\text{-}value < 0.0001$. This leads us to the conclusion that institution rank does play a role in the admission decision. Yet another interesting observation is that there is a comparable gain in webometrics rankings and all of the ways that we created and shuffled the rank-lists, consciously as well as randomly.

Table 8 provides gain in $F1_{admit}$ corresponding to each rank-list for some of the universities. We see that the universities show three types of behavior:

[17]Arizona State University

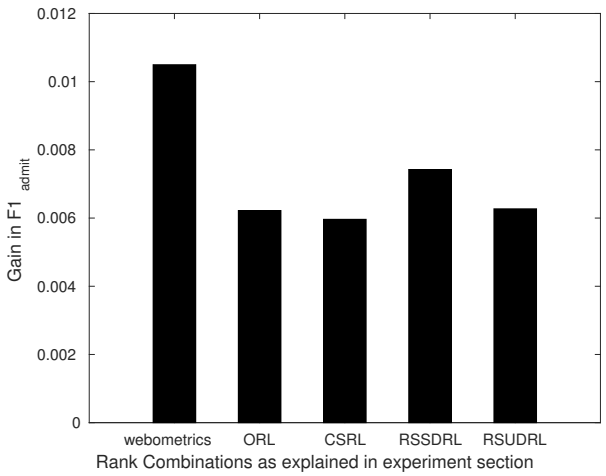

Figure 5: Gain in F1 due to various rank-lists

Table 8: Gain in $F1_{admit}$ due to various rank-lists (Average over 100 iterations) (In the order of magnitude of 10^{-3})

University	ORL	CSRL	RSSDRL	RSUDRL
ASU[17]	-0.0047	-0.1093	-0.1307	-0.2356
CMU	-0.0207	-0.7284	-0.0679	-0.2669
Brown University	0.3856	-0.8298	4.0007	-0.0333
Purdue	-2.4818	-1.4320	-1.2061	0.5790
UT Austin	5.7343	4.1302	3.5169	5.2184
Virginia Tech	4.0004	5.4014	5.7771	4.5617

1. There is loss in $F1_{admit}$ for each of the rank-lists e.g. ASU or CMU
2. Some rank-lists provide gain while others cause loss in $F1_{admit}$ e.g. Purdue or Brown universities.
3. There is gain in $F1_{admit}$ for each of the rank-lists e.g. UT Austin or Virginia Tech

A probable reason for this behavior (1) is that either the universities such as ASU or CMU don't use ranking system at all, and hence providing rank-lists causes the classifier to learn on irrelevant features causing a net loss, or none of the rank-lists is even close to the rank-lists used by these universities. For behavior (2), some of the rank-lists are close to the ones used by these universities while other rank-lists deviate drastically. For observation (3), all of the rank-lists are partially matching to the rank-lists used by these universities. Also, from 3^{rd} and 4^{th} rows in Table 8, it can be seen that one rank-lists provides gain to a specific university, while the other does the same for some other university. Hence, it can be claimed that although the universities use some form of rank-list, there is no consensus over what this actual list is.

3.7 Impact of Change in Application Year

In this experiment, we asked the question - Do universities change taste of students over time? Hence, we explored the change in decision to an application in a different application

year. Some assume that since there is an increment in the number of applications every year, admissions become more competitive over time. We performed a carefully controlled experiment to test the validity of this hypothesis. In this setting, for every university:

1. Choose a training set (80%) and test set (20%), by random selection.
2. For each record in test set, record the application year, admission decision and prediction of classifier.
3. For each record in test set, change the application year (choose randomly between 2001 and 2015), and record the new prediction, using the classifier used in the previous step.
4. Perform this experiment for n(=100) iterations.

Our hypothesis was that if yearly factors do not have an effect then, changing application year should not change the decision.

The experiment is aimed at testing the hypothesis that a change in the application year leads to a change in the decision. If an application was correctly classified initially then if there isn't any change in the competition of the application pool then it should still be correctly classified. Over 100 iterations, approximately 60K records were tested, out of which 55K were classified correctly. It was observed that out of these, a vast majority (>98%) retained the same label even after the change of application year.

We define competition per application to increase if increase in application year changed the decision from *Admit* to *Reject*, and vice versa. Out of all of the decision changes, a record was assigned '+1' if it showed that the competition increased, and '-1' if the competition decreased. Overall sum of these scores for most of the universities, individually, was very close to 0. Whereas, for all of the universities put together, the net sum was -56, which means 56/55K, approximately 0.1% decrease in competition. Hence, it can be said that decision for an application depends solely on the university and the application, and not the application year.

3.8 Which Universities Go Together

One of the unique features of our dataset construction is that applicant records capture various university combinations that the users apply to along with their results. This allowed us to find patterns, and formulate similarities among universities. Apriori algorithm [7] produces interesting results that are reported in table 10. But Apriori favors heavily populated universities over the less frequent ones. Hence, we expanded our experiments to include null-invariant measures. We computed similarity of two universities based on candidate acceptance using several null-invariant measures such as: AllConf, Jaccard, Cosine Similarity, Kulczynski coefficient, MaxConf defined in [16]. Results of the experiments are reported in Table 9 and 10.

As discussed in Section 1, applicants have the tendency to apply to universities in buckets *(Dream, Reach, Safety)*. This leads to the hypothesis that since applicants apply in buckets, it should be apparent through the similarity scores of the universities. As the results show this is infact the case. Kulczynski coefficient represents the average of conditional probability conditioned on each of the variables. Table 9 lists a few interesting associations based on the kulczynski coefficient and Table 10 lists such results for Apriori algorithm.

Table 9: Interesting similar universities based on Kulczynski score

University 1	University 2	Kulc
UChicago[18]	CSU[19]	0.286
UNCC[20]	UNLV[21]	0.303
CalTech[22]	UCR[23]	0.521
URI[24]	UWisc[25]	0.508

4. RECOMMENDATIONS

Since, now we have a system that can predict application decisions for a university, we can utilize it to aid students in making informed choices. In Section 3.8, we provide evidence that students apply to universities based on their notion of 'Dream', 'Reachable' and 'Safe' buckets. We include this notion into our algorithm to generate recommendations. Since the classifier system we have is not 100% accurate, it can generate erroneous recommendations if we simply classify for each university and return the results. Fig 3 shows that although the trend in university ranks is not strictly monotonous, it becomes very smooth if we cluster neighboring universities and then plot it. Hence, we cluster universities and employ multi-level classification to produce robust results while generating recommendations.

The first level of decision is coarse and the next level result is fine-grained. While classifying coarsely over a range of universities, we mix the records of all universities inside a cluster and train a single classifier on all of them. If the universities in the cluster are similar, the classifier learns the common patterns of admission versus rejection and provides a more general decision than any of the component universities. While in Fine-grained classification an individual classifier is trained for each university.

This algorithm consists of 5 steps:

1. **University clustering**
   - Cluster similar universities together based on US News rankings e.g. Universities in rank [1,10] fall into cluster 1, universities from [11,20] fall into cluster 2 and so on.
2. **Coarse classification**

[18]University of Chicago
[19]Chicago State University
[20]University of North Carolina Charlotte
[21]University of Nevada Las Vegas
[22]California Institute of Technology
[23]University of California Riverside
[24]University of Rhode Island
[25]University of Wisconsin Madison
[26]University of Minnesota twin cities
[27]State University of New York Stony Brook
[28]University of Illinois Chicago
[29]Indiana University-Bloomington
[30]State University of New York Buffalo
[31]George Mason University

Table 10: Universities that go together based on Apriori algorithm

University 1	University 2	Support
UM Twin[26]	SUNY Stony[27]	218
UIC[28]	Indiana[29]	112
Cornell University	SUNY Stony	97
SUNY Buffalo[30]	GMU[31]	75

- Using coarse decision, we find the cluster that offers *Admit* and is closest to the top-tier universities. We call this cluster as 'Reachable' because it is the best ranked university cluster that can offer Admission.

3. **Reachable Universities**

   - Perform fine-grained classification on each of the universities in 'Reachable', and return those which produce an *Admit* with highest probability.

4. **Safe Universities**

   - We call the cluster next to 'Reachable' as 'Safe' because it also offers admission and does so with higher probability. Then fine-grained classification is applied on 'Safe' to report Safe universities.

5. **Dream Universities**

   - For 'Dream' universities, we find those universities which are similar to the ones produced by 'Reachable' and 'Safe' but are towards the top tier universities and hence do not offer admission. These similar universities are based on the higher similarity score based on common admissions.

In step 1, the benefit of using US News rankings is that as the universities get closer to top rank, probability of admission of any candidate decreases. We also verified the same observation from data. As a future work, there many clustering schemes can be employed here, including university similarity scores reported in section 3.8, as long as proximity of various clusters is known.

5. DISCUSSION AND FUTURE WORK

This paper studies the graduate admission process in American universities using a machine learning approach. Our goal is to build a decision support model that allows candidates to make informed decisions on which schools to apply to, what are their chances of admission, and a slew of other decision-related issues. We modeled the decision process as a learning problem and presented a system that can achieve high accuracy and can be generalized across multiple universities. By employing many approaches towards solving this problem, such as supervised learning and latent variable based generative modeling, we prove that a mixture of approaches can provide better results than any of the individual approaches. We also provide a dataset that provides avenues for further research. This work can be extended in multiple ways such as towards improving accuracy or validating common notions. Some of the additions of this work may include expanding the EM formulation by modeling further variables such as undergraduate institution ranking mechanism. We proved that every university has a custom ranking mechanism. This mechanism can be modeled as a distribution which can, then, be assigned to one of the hidden variables in the EM model. Theoretically, such a model has more expressive power and can, thus, learn better regarding application decisions. We believe this is but a brisk start to the research that can be performed on the topic. Many more enhancements are possible by expanding the dataset and extracting richer data features from Letters of Recommendation or Statement of Purpose. By asking these questions and providing this dataset, we hope to initiate a discussion that can lead to better understanding of how academia accepts its new members.

6. REFERENCES

[1] Academic Ranking of World Universities. *Web*, 2015.

[2] List of Institutes of National Importance in India. *Web*, 2015.

[3] Ranking Web of Universities. *Web*, 2015.

[4] The Grad Cafe. *Web*, 2015.

[5] University Rankings | Top Universities. *Web*, 2015.

[6] US News Best Graduate Schools. *Web*, 2015.

[7] R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. *Proc. 20th int. conf. very large data bases, VLDB*, 1215:487–499, 1994.

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and regression trees. wadsworth. *Belmont, CA*, 1984.

[10] T. H. Bruggink and V. Gambhir. Statistical models for college admission and enrollment: A case study for a selective liberal arts college. *Research in Higher Education*, 37(2):221–240, 1996.

[11] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[13] B. Efron and R. Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statist. Sci.*, 1(1):54–75, 02 1986.

[14] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[15] A. Grove and D. Roth. Linear concepts and hidden variables. *Machine Learning*, 42(1/2):123–141, 2001.

[16] J. Han, M. Kamber, and J. Pei. 6 - mining frequent patterns, associations, and correlations: Basic concepts and methods. In J. H. Kamber and J. Pei, editors, *Data Mining (Third Edition)*, The Morgan Kaufmann Series in Data Management Systems, pages 243 – 278. Morgan Kaufmann, Boston, third edition edition, 2012.

[17] S. Kassegne. Edulix - premier site for scholars - 'education crowdsourced'. *Web*.

[18] S. Krishnamoorthy. Acceptance rates, apparently, are poor predictors of getting in. *The New York Times*, Apr 2013.

[19] J. S. Moore. An expert system approach to graduate school admission decisions and academic performance prediction. *Omega*, 26(5):659 – 670, 1998.

[20] K. V. S. Murthy and S. L. Salzberg. *On growing better decision trees from data*. PhD thesis, Citeseer, 1995.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[22] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[23] K. Raghunathan. Demystifying the American Graduate Admissions Process. *Web*, 2010.

[24] A. J. Smola and B. Schölkopf. A tutorial on support vector regression, 2004.

[25] A. Waters and R. Miikkulainen. Grade: Machine learning support for graduate admissions. In *Proceedings of the 25th Conference on Innovative Applications of Artificial Intelligence*, 2013.