Machine Learning Based Graduate Admission Prediction

Tianbao Li
Department of Computer Science
University of Toronto
Toronto, Ontario M5S 2E4
tianbao@cs.toronto.edu

Abstract

Graduation application to the US, Canada and the UK is a hot topic for students around the world, especially in China during the past years. Admission committee always gives out a final decision according to the student's background in many perspectives. Such decision can be considered as a classification problem and there is no much work on it by now. In this project, we applied several machine learning algorithms on our self-built dataset and trained models with relatively high accuracy.

1 Introduction

Nowadays, more and more Chinese students start to take graduate study overseas, usually in the US, Canada, the UK and somewhere else. Then, the problem just comes with it. How could they know whether a graduate school is going to offer admission or which one is the best fit?

Students usually finish their graduate application in two ways. The first one is to find an agency for help. Such agencies collect application data for year and give advice based on history cases and percentage on multiple indicators. However, misjudging always happens just because graduate admission consists of complex evaluation in many areas. The second type of application is called DIY-application, finished by students themselves. Due to lack of information, many students lose better offers.

During the application evaluation, information in many fields of one student is considered, including TOEFL score, GRE score, GPA and and other supplementary materials as undergraduate school, work experience, research experience. Looking in the machine learning way, these indicators can be features of a certain model. We can use massive admission and rejection cases as training data to fit the admission model of a certain graduate program. Till now, decision fot daily useage is mainly made by human experience in this area.

Comparing to human judgement and finding similar cases, many machine learning models seem to have a better prediction. A few related research built the process of decision making. [5] and [1] give out the qualitative influence measurement on each component of the application material. Some work such as [6] is used to offer decision making assistance for universities. Work like [2], [4] and [3] use well-distributed data to build a statistic model and get good result. However, work like ours aims at producing a general prediction but not on a cetrain school. It means more valuable for applicants with multiple school choices. Also, the high quality data with great distribution that they use for certain universities is hard to get.

The contribution of this project is mainly is these points:

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

- Dataset built-up: here I decide to use data from the BBS GTER[1], one of the most popular graduate application BBSs in China. Many Chinese students post their admission decision here. I decide to write crawler to collection data of admission and rejection. Also due to the low data quality, much work on data cleaning has to be done. Such data can be used for related purpose in the future research

- Model training: here I decide to train several popular machine learning models on such data, including neural networks, decision tree, naive bayes, etc, find better fit model and make optimization.

Clarification

This project is finished by myself alone. With a self-defined dataset, some related work may continues on it afterwards.

2 Data Summary

Chinese students take a large portation in graduate application, however, there is no avaliable dataset about it. So, for the first part of this project, we intent to biult a dataset about Chinese student graduate application. We wrote web crawler to collect application data from GTER BBS, both admission and rejection application in the range of 2012-2016. Since web data usually holds low quality, some data cleaning skills are applied on the dataset to make it easy for model training.

Current dataset contains 11056 cases. Due to natural language-based expression, the raw data is hard to catch related feature. Here, we only extract information about decision, target school, degree, year of application, TOEFL[2] score, GRE[3] score, GPA, GPA ranking. Such information is transformed into 14 features and well normalized for training. Summary of the dataset is shown in Table 1.

General Information	
data point amount	11056
feature amount	14
Features for dataset	
result	admission:reject=4.48
most popular school	Columbia University
year range	[2012,2015]
TOEFL total average	84.8
TOEFL reading average	25.2
TOEFL listening average	22.1
TOEFL speaking average	23.1
TOEFL writing average	26.6
GRE average	319.0
GPA average	3.2

Table 1: Data summary

3 Prediction Models

To make a prediction whether a student will be admitted or rejected (binary classification), several models are trained and compared to get a relatively best model. From the direct perception of the data, in my opinion, decision tree should work well for the problem containing criterion of judgment, also KNN for same class data gathering together. So I also use a model simply conbining decision tree and KNN together and try to get a promotion on accuracy. In the following section, we use y

[1] `http://bbs.gter.net/`
[2] `https://www.ets.org/toefl`
[3] `https://www.ets.org/gre`

to represent class variable and $\{x_1, x_2, \dots, x_n\}$ for features. All the models are implemented by scikit-learn[4].

3.1 Naive Bayes

Naive Bayes is a generative model based on Bayes' theorem. Furthermore, it gives an assumption on features independency. It means that

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y)$$

so the decision rule is that

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y) \prod_{i=1}^{n} P(x_i|y)$$

For implementation, we use MultinomialNB[5] from scikit-learn. To train a better fitting model, alpha (additive (Laplace/Lidstone) smoothing parameter) can be change as hyper-parameter.

3.2 Logistic Regression

Logistic regression is a linear classification model. Commonly, the model is optimized by minimize the loss function (take l2 penalty as an example)

$$min_{w,c} \frac{1}{2} w^{\mathrm{T}} w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^{\mathrm{T}} w + c)) + 1)$$

For implementation, we use LogisticRegression[6] from scikit-learn. In this model, hyper-perameters such as penalty (l1 or l2), C (inverse regularization strength), tol (tolerance for stopping criteria) and some others can be changed.

3.3 SVM

Support Vector Machine is also a popular linear classification method with good performance. However, due to its advantages in high feature dimension, it is not expected to work so well for our scenario.

LinearSVC[7] from scikit-learn. C (penalty parameter of the error term) is being monitored as hyper-parameter.

3.4 KNN

Distance-based K Nearest Neighbors is one suitable model for our problem. The first thought is that admitted cases hold the similar value on features, which means they are near in the feature space. It just fits the idea of nearest neighbors and it tends to work well.

For KNN, we use KNeighborsClassifier[8] from scikit-learn as the model. Then we try to justify the hyper-parameter K (amount of neighbors) to have a higher accuracy.

---

[4] http://scikit-learn.org/stable/

[5] http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

[6] http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[7] http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[8] http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

3.5 Decision Tree

During the human admission process, it is usually divided into several decision rules, such as if the GPA high enough or if English using hits the criteria. Such process is similar to the decision tree classification. Also, the representative capability increases with the tree comes deeper.

In scikit-learn, DecisionTreeClassifier[9] can be used in this problem. For the hyper-parameter max_depth, it could overfit when larger and underfit when smaller. So the parameter adjustment can be applied on this.

Futuremore, researchers usually use decision trees with much randomness, that is random forest. Comprised of several different decision trees, it often has better usually results on complex problem.

3.6 Neural Networks

As the most popular model, neural networks is solving quite different problems, linear or non-linear, simple or complex... Our project also play a easy try on this. MLPClassifier[10] is a non-GPU version implementation for small-scale application. Hyper-parameters of hidden layer sizes, activation functions are L2 penalty (regularization term) parameter are adjusting to achieve better.

3.7 Combination of KNN and Decision Tree

Following the discussion above, KNN and decision tree are the two models that could work well. To hold both advantages, here, we use a simple way to combine KNN and decision together (KNN-Tree).

First we train models of KNN and decision tree and get the class label as y_{knn} and y_{tree}. Then a weighted sum (with α) and threshold comparison (with β) work together to get the final label. That is

$$y = \begin{cases} 1, & \alpha y_{knn} + (1-\alpha) y_{tree} > \beta \\ 0, & \alpha y_{knn} + (1-\alpha) y_{tree} \leq \beta \end{cases}$$

α and β are selected as hyper-parameters.

4 Experient Evaluation

To evaluate the existing and proposed models, we try to use the self-built dataset and predict whether a student can be admitted by a university. To make a better use of the data, at first, we run data cleaning and normalization on the each feature and then calculate the accuracy for each model. Detailed process is introduced as follows.

4.1 Data Cleaning and Normalization

As out dataset is captured from web with human input, their should be a lot of errors or format inconsistence. To make use of it, the first thing to do is to clean the data and transform the information to the feature that we need. Also, normalizing the data into a similar range can help models work better.

Data cleaning on this dataset contains:

- natural language extraction: there is some information in the inexplicit natrual language words as admission result. Some rules are written for this.
- inconsistence: some information are inconsistence among cases like university name. Regularization work lile string fuzzy matching is needed for this. Here we use python package FuzzyWuzzy[11] and match user input with common-used university name.

[9] `http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`

[10] `http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html`

[11] `https://github.com/seatgeek/fuzzywuzzy`

- missing value filling: due to some reson, some cases lack of TOEFL or GRE value. For keeping the influence to a lowest range, we fill them with average value for each.
- data format: the raw data is in csv format, with much comma as a part of values. The data is well-split and transformed into json file finally.

Data normalization on thie dataset contains:

- university: university name is a non-numerical value and hard for training. we use university ranking from Times[12] and replace the university name. Normally, highly ranked universities have high admission requirement.
- degree: 1 for master and 2 for Ph.D
- application year: map year [2012,2013,2014,2015] into[0.00,0.25,0.50,0.75]. (leave 2016 to 1 behind for future extension)
- decision result: 0 for rejection and 1 for admission (only admission and offer)
- TOEFL score: change total score [0,120] into [0,1] and sub-score from [0,30] into [0,1] by division.

$$total\_normalized = total/120$$
$$sub\_normalized = sub/30$$

- GRE score: change total score [260,340] into [0,1], Verbal/Quantitative score from [130,170] into [0,1] and writing score from [0,6] into [0,1] by division.

$$total\_normalized = (total - 260)/80$$
$$sub\_normalized = (sub - 130)/40$$
$$writing\_normalized = writing/6$$

- GPA: most cases have GPA score in the form of 4.0-total or 100-total. Just turn them into [0,1] by official method.
- GPA ranking: transforming into percentile

4.2 Evaluation Metrics

Evaluation used in the project is one of the most simple by effictive, accuracy. It is shown as this:

$$accuracy = \frac{hits}{total}$$

4.3 Model Comparison

For model comparison, we split the dataset into training data and test data, which test data holds the precentage of 20%. Also for model training, 20% of training data is for validation and hyper-paramater choosing.

The detailed train accuracy, test accuracy are shown in Table 2.

From the result of the accuracy, best perfoemed existing models are KNN Decision Tree and Random Forest. What's more, out proposed combination model of KNN and Decision Tree improves the accuracy by 2-3 percentage.

References

[1] Maha Alzahrani. *" I Got Accepted": Perceptions of Saudi Graduate Students on Factors influencing their Application Experience*. PhD thesis.

[2] Thomas H Bruggink and Vivek 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.

[12] `https://www.timeshighereducation.com/world-university-rankings`

Model	Train accuracy	test accuracy
MultinomialNB	0.568401371144	0.549190535492
Logist Regression	0.620442505453	0.683483802443
SVM	0.620442505453	0.626400996264
KNN	0.998753505765	0.765877957659
Decision Tree	0.999065129324	0.752179327522
Random Forest	0.999065129324	0.753424657534
Neural Networks	0.616703022749	0.59900373599
KNN+Decision Tree		0.789539227895

Table 2: Model comparison

[3] Narender Gupta, Aman Sawhney, and Dan Roth. Will i get in? modeling the graduate admission process for american universities. In *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*, pages 631–638. IEEE, 2016.

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

[5] Karthik Raghunathan. Demystifying the american graduate admissions process. *StudyMode.com*, 2010.

[6] Austin Waters and Risto Miikkulainen. Grade: machine learning support for graduate admissions. *AI Magazine*, 35(1):64, 2014.