Machine Learning Engineer Nanodegree

Predict College Earning Potential

Aloysius Joseph

July 15, 2018

I. Definition

I.1 Project Overview

I propose to create a model for prediction for college selection based on earning potential. Students and parents have a tough time determining which colleges to apply. There are a lot of factors to consider and lots of conflicting information. Also there is lots of data available as well as lots of variables involved. But in general apart from SAT score and GPA that are used mainly for the admission process, several factors like University admission rate, public/private type of university etc., need to be considered.

Dataset:
https://collegescorecard.ed.gov/data/Most-Recent-Cohorts-All-Data-Elements.csv

Documentation:
https://collegescorecard.ed.gov/assets/FullDataDocumentation.pdf

Data Dictionary:
https://collegescorecard.ed.gov/data/CollegeScorecardDataDictionary.xlsx

I.2 Problem Statement

Predict some of the major factors to be considered by a student when applying to Universities and help in the process of selecting Universities to apply.

This is a classification problem. If the student is hoping to earn at least $50,000 10 years after graduation, which universities might he plan on applying (without taking the degree major into consideration). Which of these factors available about Universities will matter most: SAT scores, size (number of students), spending per student by university, type (public/private), cost for students, rate of admission, rate of completion and rate of retention. I plan to use Ensemble methods like Ada Bost, Random Forest and Gradient Boost and select the one providing better accuracy.

I.3 Metrics

Accuracy measures how often the classifier makes the correct prediction. It's the ratio of the number of correct predictions to the total number of predictions.

Precision is a ratio of true positives (students classified as earning >=50k, and who are actually earning that much) to all positives (all students classified as earning >=50k, irrespective of whether that was the correct classification).

`[True Positives/(True Positives + False Positives)]`

Recall(sensitivity) is a ratio of true positives(students classified as earning >=50k, and who are actually earning that much) to all the students who were actually earning >=50k.

`[True Positives/(True Positives + False Negatives)]`

For classification problems with distributions like in our case, precision and recall come in very handy. These two metrics can be combined to get the F1 score, which is weighted average of the precision and recall scores. This score can range from 0 to 1, with 1 being the best possible F1 score. The weight β can be increased if we want to have more emphasis on precision.

$$[F\beta=(1+\beta 2)\cdot precision\cdot recall/(\beta 2\cdot precision)+recall]$$

II. Analysis

II.1 Data Exploration

The dataset is provided by the US Department of Education (https://collegescorecard.ed.gov/). It is provided as a CSV file. The dataset consists of approximately 7593 data points, with each datapoint having 1825 features.

But for the purpose of this project 9 features have been selected with 1 target variable, as these seem to be more appropriate for the problem at hand.

Features

- `CONTROL`: integer : Collge type (1-Public/2&3-Private)
- `ADM_RATE`: float : Admission Rate
- `SATVRMID`: float: Average SAT score in English
- `SATMTMID`: float : Average SAT score in Math
- `COSTT4_A`: integer : Average cost to complete education
- `C150_4`: float: The degree completion rate
- `RET_FT4_POOLED`: float: Student retention rate
- `NUM4_PUB`: integer : Total number of enrolled students (indicates size)
- `INEXPFTE`: integer: Instructional expenditure per student

Target Variable

- `MN_EARN_WNE_INC2_P10`: float: Mean earnings of students 10 years after entry (<=50K, >50K)

II.1.1 Data Sample:

	UNITID	OPEID	OPEID6	INSTNM	CITY	STABBR	ZIP	ACCREDAGENCY	INSTURL	NPCURL
0	100654	100200	1002	Alabama A & M University	Normal	AL	35762	Southern Association of Colleges and Schools C...	www.aamu.edu/	www2.aamu.edu/scripts/netpricecalc/npcalc.htm
1	100663	105200	1052	University of Alabama at Birmingham	Birmingham	AL	35294-0110	Southern Association of Colleges and Schools C...	www.uab.edu	uab.studentaidcalculator.com/survey.aspx
2	100690	2503400	25034	Amridge University	Montgomery	AL	36117-3553	Southern Association of Colleges and Schools C...	www.amridgeuniversity.edu	www2.amridgeuniversity.edu:9091/
3	100706	105500	1055	University of Alabama in Huntsville	Huntsville	AL	35899	Southern Association of Colleges and Schools C...	www.uah.edu	finaid.uah.edu/
4	100724	100500	1005	Alabama State University	Montgomery	AL	36104-0271	Southern Association of Colleges and Schools C...	www.alasu.edu	www.alasu.edu/cost-aid/forms/calculator/index....

5 rows × 1825 columns

II.1.2 Data Statistics for selected features and target:

	Earning	NoStudents	UnivSpending	UnivType	SATMath	SATRead	StudentCost	AdmissionRate	Completionrate	RetentionRate
count	448.000000	448.000000	448.000000	448.0	448.000000	448.000000	448.000000	448.000000	448.000000	448.000000
mean	47238.392857	1038.589286	9676.296875	1.0	512.169643	524.897321	21166.167411	0.682814	0.511222	0.768067
std	8848.756487	805.681157	4378.046492	0.0	55.772049	62.385990	4059.443911	0.168965	0.170033	0.096141
min	29800.000000	54.000000	3121.000000	1.0	370.000000	380.000000	12151.000000	0.168836	0.108900	0.488141
25%	41100.000000	387.500000	7039.750000	1.0	475.000000	485.000000	18452.500000	0.582301	0.397175	0.703396
50%	46200.000000	828.500000	8591.000000	1.0	505.000000	515.000000	20949.500000	0.690453	0.493250	0.766676
75%	51300.000000	1476.750000	10817.500000	1.0	545.250000	555.000000	23340.000000	0.805161	0.632250	0.840963
max	95900.000000	3975.000000	43996.000000	1.0	680.000000	745.000000	34496.000000	1.000000	0.933000	0.972121

II.1.3 Data Sample for selected features and target:

	Earning	NoStudents	UnivSpending	UnivType	SATMath	SATRead	StudentCost	AdmissionRate	Completionrate	RetentionRate
0	35500.0	743.0	7941.0	1	427.0	420.0	20809.0	0.653841	0.3081	0.616384
1	45900.0	955.0	17548.0	1	575.0	594.0	22232.0	0.604275	0.5462	0.807657
3	53400.0	331.0	10619.0	1	585.0	615.0	20999.0	0.811971	0.4935	0.786986
4	30700.0	570.0	7742.0	1	410.0	410.0	18100.0	0.463858	0.2696	0.584708
5	50100.0	1282.0	10312.0	1	545.0	550.0	27205.0	0.535867	0.6709	0.865822

II.1.4 Final DataSet details:

Some of the data records did not have information for Earnings from Universities as it was privacy protected. Hence had to pre-cleanup to remove those records.

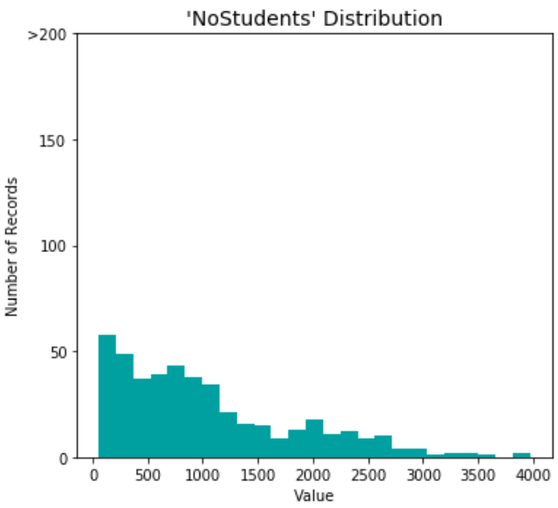
```
Total number of records: 448
Individuals making more than $50,000: 150
Individuals making at most $50,000: 298
Percentage of individuals making more than $50,000: 33.48%
```

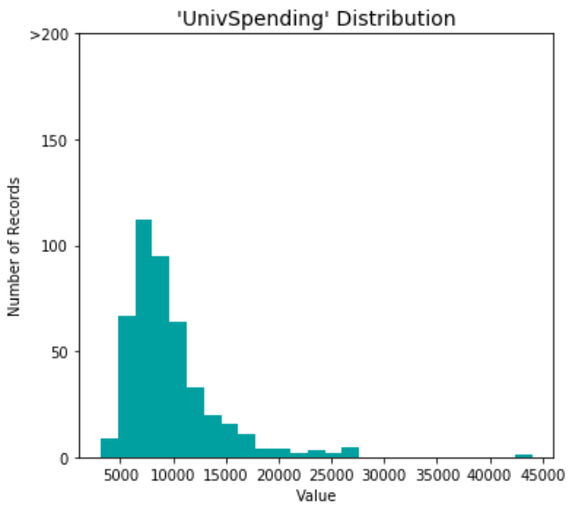
II.2 Exploratory Visualization

The various distributions of the Features are displayed below:

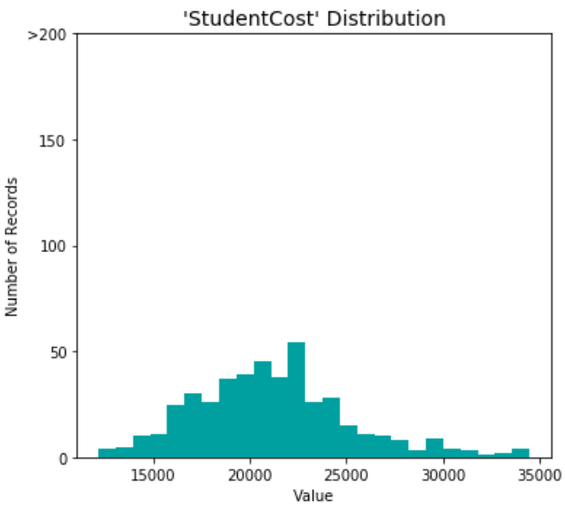
Observations:

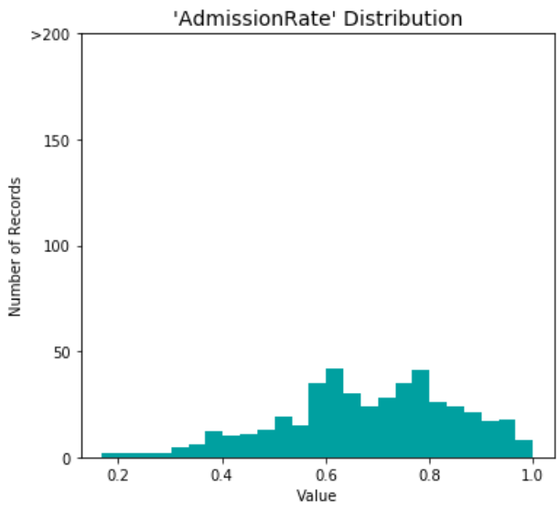
The distributions for number of students does not have a normal distribution and skewed to left, as there are large universities and small private colleges. The distribution for university spending on students is highly skewed to the left, as few famous universities have huge endowments while most others do not have that type of funding to spend on students. Hence we may have to do log-transformation for these features so they do not negatively affect the performance of a learning algorithm.

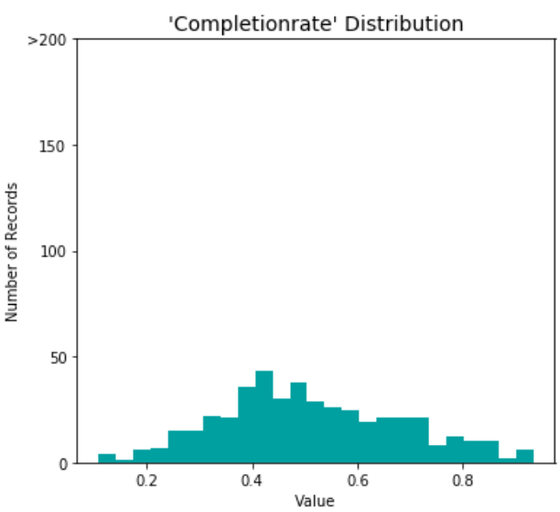


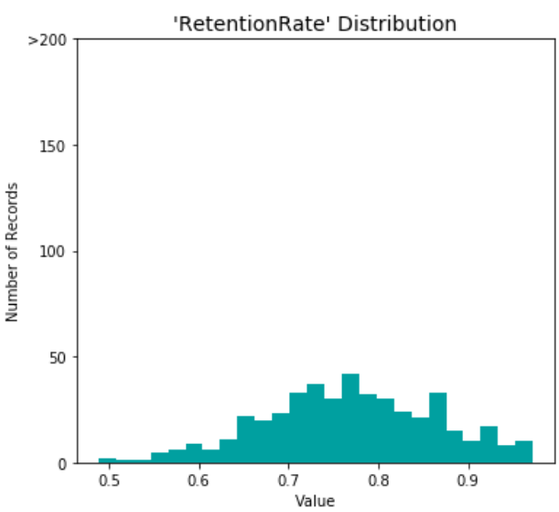


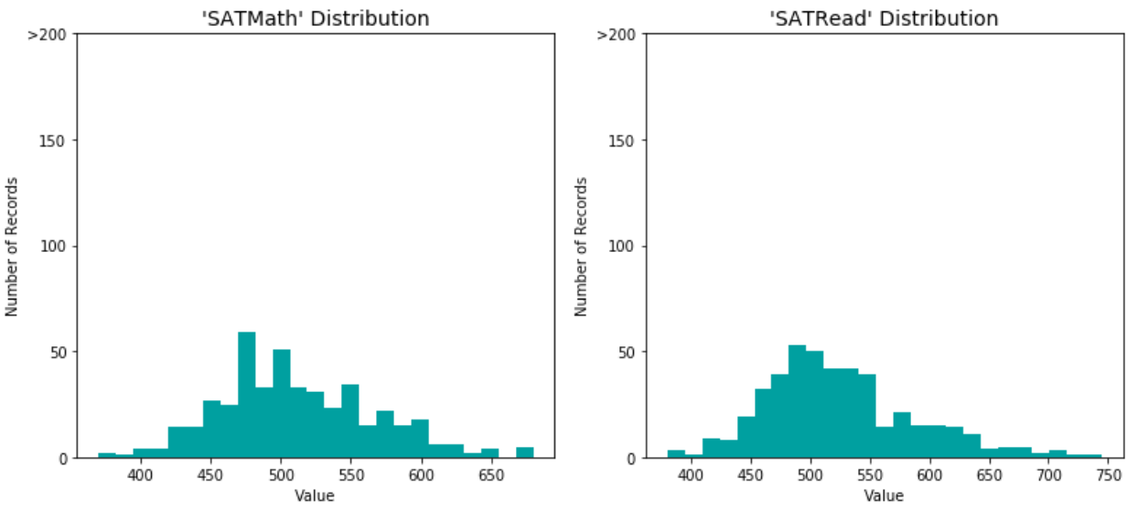
The distribution of other features namely student cost, admission rate, completion rate, retention rate and SAT scores generally have somewhat normal distributions which might be good enough for this prediction.



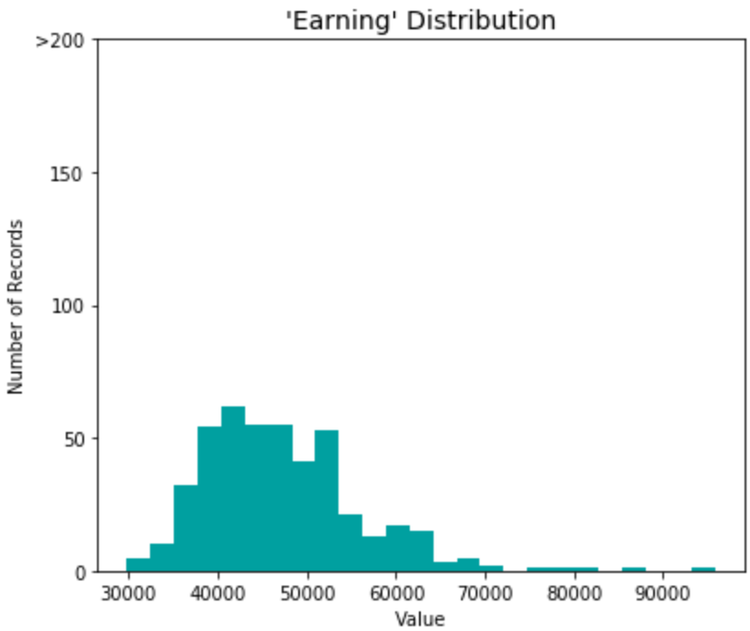








The distribution for the target Earning variable is shown below. It is skewed to the left as more students earn less than 50K as we saw in the statistics section above.



II.3 Algorithms and Techniques

This is a Classification problem as we are trying to predict if students will earn at least 50K after 10 years of graduation based on a selected feature set. A decision Tree based model could be used as a starting point and Ensemble methods below can be attempted.

Based on http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html and http://scikit-learn.org/stable/modules/ensemble.html I am thinking of using Ensemble Methods sequentail boosting (AdaBoost, Gradient Boost) and parallel bagging(RandomForest). This is because these classifiers make use of a base classifier (here Decision Tree as default) and improve on that. Error = Bias (where the algorithm cannot learn the target) + Variance (comes from sampling)

RandomForest (Bagging): It is based on fully grown decision trees (low bias, high variance). Bagging reduces error mainly by reducing variance (but not bias) by making the trees uncorrelated. The main weakness is it needs fully grown trees hence increases computational complexity of the model. Can be slow to score as the complexity increases. Its main strength is ability to handle outliers and noise. Also it works fast and off the shelf and typically avoids overfitting. This is a good model for the data as the dataset is not too large or complex and is generally considered a safe bet.

AdaBoost, GradientBoost (Boosting): Boosting is based on weak learners (high bias, low variance).Boosting reduces error mainly by reducing bias (and also to some extent variance), by aggregating the output from many models. Its main weakness is inability to handle outliers and noise and can overfit. It is complex to do tuning with several hyperparameters and finding a stopping point, But it performs well with higher complexities and is fast. They give better results but much harder to train. These are good models for the data as they are very powerful and as the data has been preprocessed for outliers. I guess GradientBoosting is used in most winning Kaggle competitions for a reason.

GradientBoost can be thought of a specific type of AdaptiveBoosting. The main difference is in AdaBoost boosting is done by increasing the weight of incorrect observations so subsequent iterations concentrate on those, whereas in GradientBoost gradient descent logic is used to minimize the loss function when adding trees. In a sense GradientBoost is generic enough to be used for more situations than AdaBoost.

I am thinking of starting with the default parameters for these algorithms, and then fine tune them later to see if they make a difference in accuracy scores.

II.4 Benchmark

Since there are numerous factors that could be used and hence not many standardised studies are available to provide an historic benchmark, I am using Naive predictor methodology where we assume everyone earns above $50,000, as a benchmark model and see how the above Ensemble models perform.

The scores I got based on his method:

Naive Predictor: [Accuracy score: 0.3348, F-score: 0.3862]

III. Methodology

III.1 Data Preprocessing

The dataset had 1825 features. Fortunately there was extensive documentation, which I had to read through to shortlist the features needed.

Next some of the data records did not have information for the target variable, Earnings, as it was privacy protected. The value was mentioned as "PrivacyProtected". So I had to drop those records.

Some records had several values for the features as invalid (NaN).Had to dop those records to make sure the analysis was close a possible to real data.

The distributions for number of students and university spending on students was highly skewed to the left. Hence had to do log-transformation for these features so they do not negatively affect the performance of a learning algorithm. Using a logarithmic transformation significantly reduces the range of values caused by outliers.

Also applied normalization for the features, since they were numerical, so that each feature is treated equally when applying supervised learners

Finally since I wanted to frame this as a classification problem, encoded the target variable Earnings to 1 if >=50000 and 0 if <50000.

III.2 Implementation

III.2.1 Shuffle and Split Data

First step is to split the data (both features and their labels) into training and test sets. 80% of the data will be used for training and 20% for testing.

III.2.2 Generate a Naive predictor

The purpose of generating a naive predictor is simply to show what a base model without any intelligence would look like. This is because I could not find any free historically similar research. In Naive predictor methodology I assume everyone earns above $50,000 and generate accuracy scores.

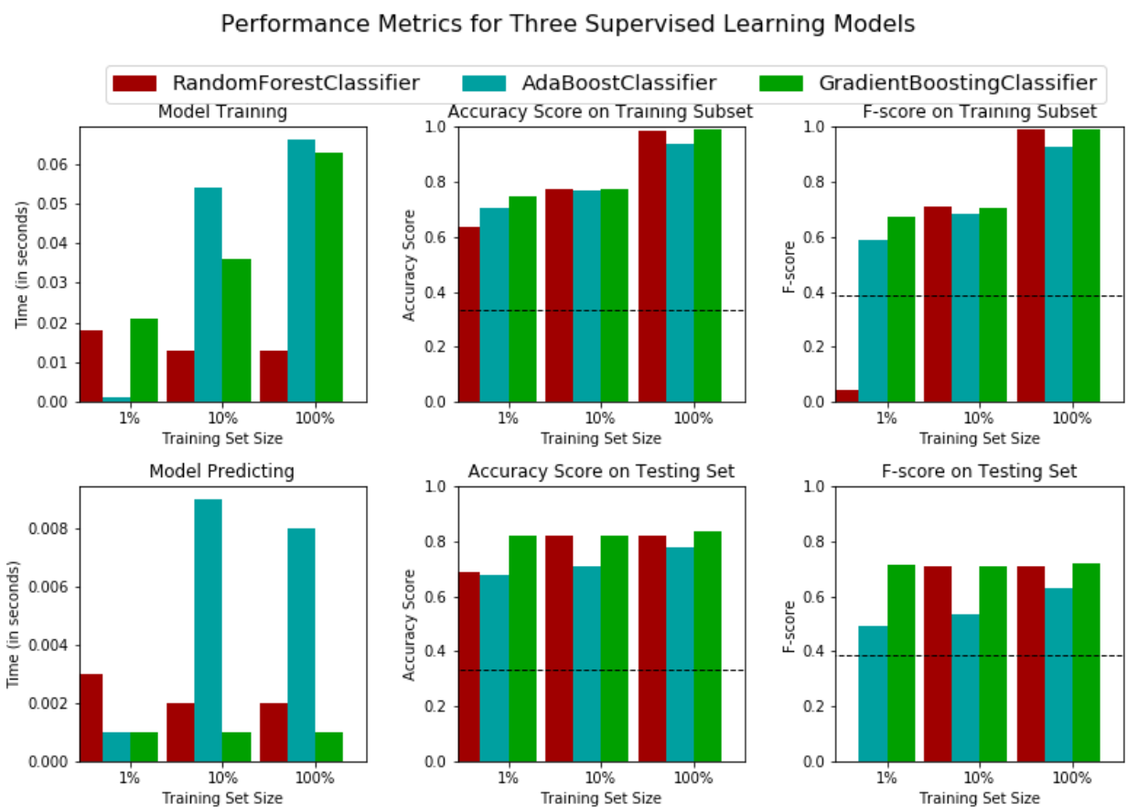
III.2.3 Select Algorithms to test

Using Ensemble Methods sequential boosting (AdaBoost, Gradient Boost) and parallel bagging (RandomForest) to compare. To properly evaluate the performance of each model, created a training and predicting pipeline that allows to quickly and effectively train models using various sizes of training data and perform predictions on the testing data.

- Fit the learner to the sampled training data and record the training time.
- Perform predictions on the test data, and also on the first 300 training points.
- Record the total prediction time.
- Calculate the accuracy score for both the training subset and testing set.
- Calculate the F-score for both the training subset and testing set.

III.2.4 Initial model evaluation

Imported the three supervised learning models as discussed. Used the default settings for each model. Calculated the number of records equal to 1%, 10%, and 100% of the training data.



Best Model : **GradientBoost** Classifier has the highest score with less time.

GradientBoost has higher F-score and Accuracy than AdaBoost in with Testing set and Training set. It also took more time than AdaBoost. Surprisingly, RandomForest did as good as Boosting algorithms in Training and took less time too. RandomForest did as good as GradientBost with slightly les accuracy scores and slightly more time during testing.

III.3 Refinement

I selected GradientBoost and fine tuned the following parameters:

```
learning_rate': [0.1, 0.05, 0.01]

n_estimators' :[25,50,70]

max_depth' : [1,3,9]
```

learning_rate : shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.

n_estimators : The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance.

max_depth : limits the number of nodes in the tree.

Unoptimized model Accuracy score on testing data: 0.8333 F-score on testing data: 0.7194

Optimized Model Accuracy score on testing data: 0.8444 F-score on the testing data: 0.7407

Final Optimized Model with Feature Selection: Finally wanted to see how the model will perform if only the top five important features were usd to train. Accuracy score on testing data: 0.8556 F-score on testing data: 0.7634

IV. Results

IV.1 Model Evaluation and Validation

The Final optimized model scores are Accuracy 0.8556 and F-score 0.7634 The Optimized model with feature selection is better than the Optimized model which is better than the unoptimized model. But it has much better scores than benchmark (Accuracy-0.3348, F-score: 0.3862)

Results:

Metric	Unoptimized Model	Optimized Model	Optimized(Feature) Model
Accuracy Score	0.8333	0.8444	0.8556
F-score	0.7194	0.7407	0.7634

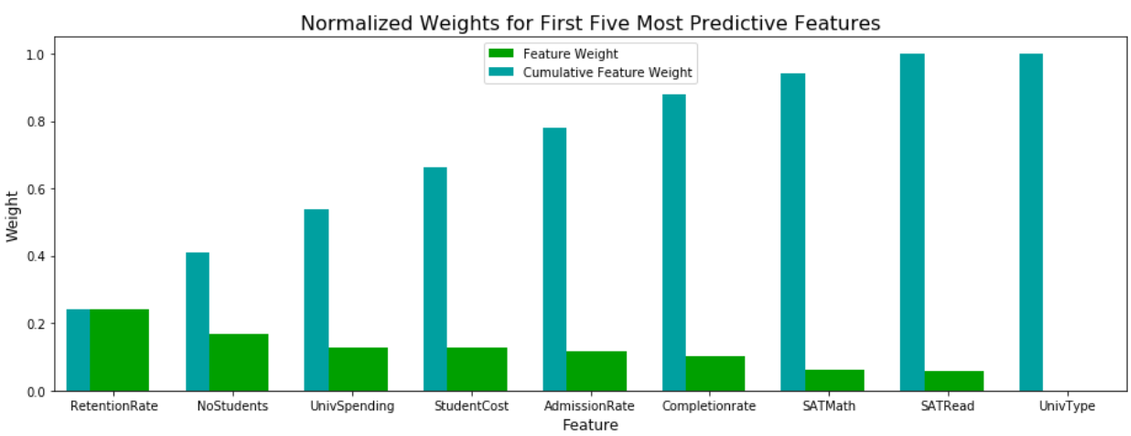
IV.2 Justification

Benchmark scores: Accuracy-0.3348, F-score: 0.3862 Final model scores: Accuracy-0.8556, F-score: 0.7634 The final model has an increase of 0.5208 in accuracy score and an increase of 0.3772 in F-score. Compared to the Benchmark the selected tuned model has gained 155.5% in accuracy score and 97.7% in F-score. Though the benchmark was a Naive model it is still impressive.

V. Conclusion

V.1 Free-Form Visualization

According to the best fitter model the importance of the features are as below:



The order of importance of the features make sense. But couple of surprises:

- College retention rate is ranked the highest as far as getting at least 50K earning in future.
- The type of University (public/private) did not seem to matter, so maybe students need not worry about high cost private colleges.

V.2 Reflection

The project can be summarized as following steps:

1. Data Exploration: Cursory investigation of the data to explore the dataset and the featureset and to find degree of correlations between variables.
2. Data Preprocessing: Cleaning the data including formatting and restructuring, normalizing the data and shuffling and splitting the data into training, validation and testing sets.
3. Feature selection: Extract feature importance, select relevant features and create new features if necessary/possible to improve accuracy.
4. Model selection: Experiment with Ensemble algorithms AdaBost, Random Forest and Gradient Boost to find out the best algorithm for this scenario.
5. Model Tuning: Fine tune the selected model to increase accuracy and performance, including restricting top 5 features.
6. Testing: Test the model on testing dataset and do final model evaluation.

Conceptually the data available for doing this type of investigation is not sufficient as Universities do not want to share this information. Also maybe there are more important factors like college major, but that is out of scope of this investigation. I think within the limitations and constraints of data, this study can help students and parents to think outside the box, to look at other factors when selecting colleges.

V.3 Improvement

To investigate further more algorithms can be tried out like XGBoost.

Can also try fine tuning other parameters with current model.

Also I had dropped records with features that did not have values. Possibly can try out with filling with average value for the feature, to have a larger dataset.

Given the problem space with scarcity of data but large interest, there is a huge potential for a business model.