Project 3

Atul Lanka, Rushil, Yuyu, Tim

December 5, 2016

1 Abstract

This project consists of performing a virtual consulting analysis from College ScoreCard data. Specifically, our team members analyzed and interpreted the data by EDA analysis, regression model building and interactive shiny app to help a group of art institution administrators to compare themselves with similar schools in California, in terms of diversity of graduation rates.

2 Introduction

Data Analysis has been extensively used in different areas to explore, understand and predict our daily life. The motivation behind this project is to provide insights for higher education officials/policymakers to better assess how well institutions are providing access to diverse group of students. Our team decides to help a group of 4-year art institution administrators to compare themselves to similar schools in California, in terms of diversity and graduation rates.

The overall workflow can be summarized as followed: data cleaning (target schools that are 4-year art institutions located in California)-¿ EDA Analysis(provide overall pictures of schools enrollment and graduation statistics)-¿Model Building()-¿ Shiny App (interactive tool for user to visualize data with selected school and other criteria)-¿Presentation slides(share results with public). Each section of the workflow will be explored and explained in details.

3 Data Cleaning

The raw data used in this project comes from College ScoreCard, downloaded, developed by the U.S Department of Education. To select targeted school, we choose specifically four-year art schools in CA (these are indicated by the abbreviation ”CCBASIC==30” and ”STABBR==”CA” respectively in the raw data). We also extracted schools' location, zip code, website etc in similar way.

There are several indicators for diversity, which we identified as gender, average family income, average age and ethnicity. For the dependent variables, we choose enrollment and graduation rates. We further divide enrollment

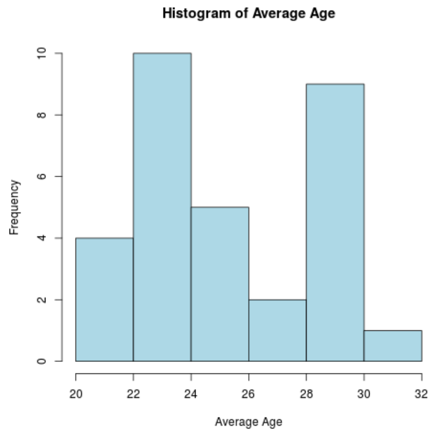

Figure 1: Average Age of Enrolled Students

Figure 1 shows enrollment rate vs average age. We can see two peaks at 22-24 and 28-30. It's expected that most people will choose to pursue a professional art degree after college, potentially as a way to change their c'areer path (majority of universities/college don't have specialized art/design program). There's also a scenario where after working for several years, people decides to go back to school either for career switch or further their knowledge and skills.

based on ethnicity (enroll-white is percentage of enrolled people that are white etc), gender(enroll-men etc), graduation based on ethnicity(grad-rate-asian etc). Note that the raw data doesn't provide graduation based on gender. We also catalog schools based on their region (1 indicates North Cal, 2 Central Cal and 3 for South Cal).

Based on the criteria, 31 schools in total are selected and the finalized data is presented in "client-dat.csv".

4 EDA Analysis

As the first step of data analysis, EDA tries to provide a general picture of average school performance. A picture is worth a thousand words, so let's take a look at what information the data discloses.

For a more quantitative data summary, we also include "eda-output-enroll" and "eda-output-grad" in the data file if readers are interested.

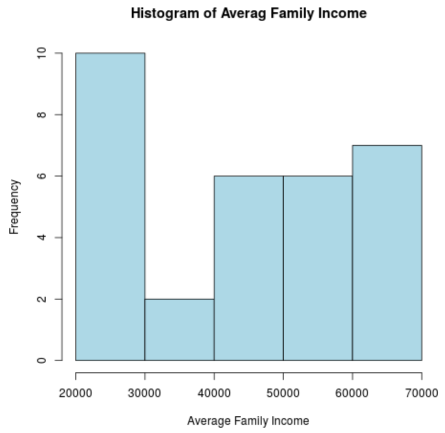

Figure 2: Average Family Income of Enrolled Students

Figure 2 shows enrollment rate vs average family income. Again we can see relatively two prominent peaks at 20-30K and 60-70K. For people on the lower end of family income, studying art/design could be a relatively lucrative path given that arts, entertainment and design are big business in California. (Hollywood down South and Berkeley/Santa Clara up North). For people on the higher end of family income, it's possible that they have been receiving high quality cultural/artsy education since young and has adequate family financial support to purse an artsy path.

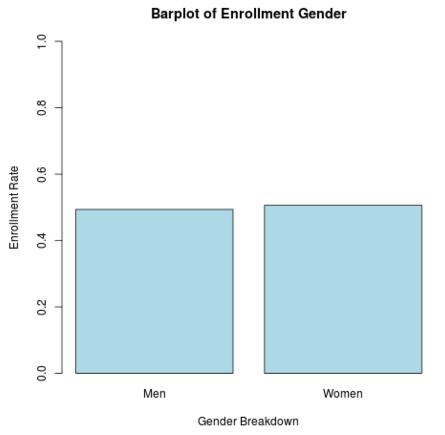

Figure 3: Gender Distribution of Enrolled Students

Figure 3 shows enrollment rate vs. gender, which simply indicates that for all 31 schools, female and male students populations are almost the same.

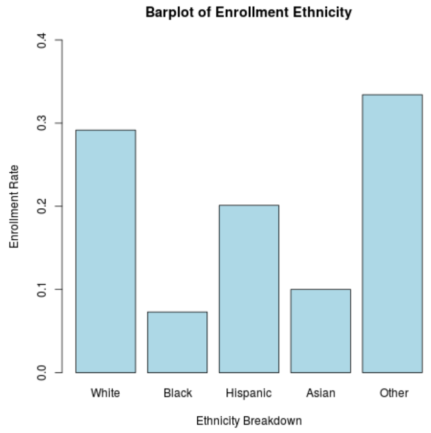

Figure 4: Ethnicity Distribution of Enrolled Students

Figure 4 shows enrollment ethnicity breakdown. The bar-plot indicates White and Hispanic make up half of the enrollment population. Since there aren't enough raw data available from College Scorecard (say ethnicity breakdown on average family income), it's difficult to extrapolate any other correlations (for example, could the 20-30K income peak corresponds to Hispanic enrollment?). Note that highest peak is "Other". This is because for simplicity, we combined all other race columns (except white, black, hispanic and asian) when producing the graphs. So "others" encompasses Indian-America, unknown, 2 or more race etc and thus the high percentage.

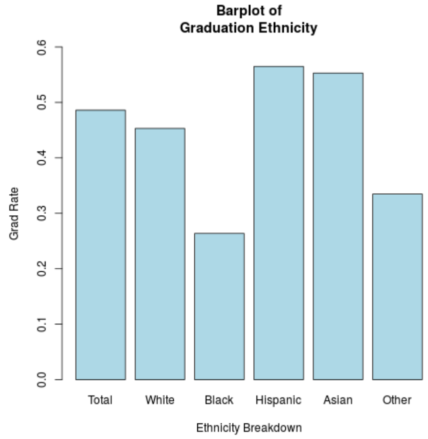

Figure 5: Ethnicity Distribution of Graduated Students

And finally, Figure 5 shows graduation rate ethnicity breakdown. Several things that need special attention. First, similar to Figure 4, the "Other" category includes multiple race options (it would be impractical to plot 12 bars into a single graph). Second, the total percentage of all bars don't add up to 1 due to nature of the clean data. For "Total" categories, it means the percentage of students that graduated within 4 years in total enrolled population. However, for "White", "Black", "Hispanic", "Asian" categories, it shows the percentage of students that graduated within their own racial category. (e.g about 45% of white students graduated within 4 years in all white enrolled students). We comment that Hispanic and Asians have more than 50% graduation rate while Black students suffer from low graduation rate. Again similar to Figure 4, due to deficiency of relevant data (e.g ethnicity vs average income), no other definitive conclusions can be drawn from the plot.

5 Methods

We built various linear models from the *client-data.csv* data set. We predicted the variable 'Grad_Rate' in terms of eight predictors: proportion of White students enrolled, proportion of Black students enrolled, proportion of Hispanic students enrolled, proportion of Asian students enrolled, proprotion of Women enrolled, Average age of enrolled students, Average Family Income and Total Enrollment.

We created 5 different scripts for running various regression methods: Ordinary Least Squares, Ridge regression, Lasso Regression, Principal Components Regression, and Partial Least Squares Regression.

Our first method, was a simple linear model, Oridnary Least Squares. In **ols_regression.R**, using the `lm()` where the y variable is Balance and the x variable is the combination of the the rest of the variables, the results from the OLS regression methods are output. This script's outputs will be used as a basis for all other regression models to compare to. We read in the scaled data set, which in turn is used for the regression model.

Our next two methods, Ridge and Lasso regression, are shrinkage methods. The script, **Ridge_regression.R**, performs a ridge regression and **Lasso_regression.R** performs a lasso regression. The steps for these two methods are nearly identical:

1. Load *scaled-client.csv*, and `library(glmnet)` and set the seed, for reproducibility sake.

2. Create a 'x' and 'y' varible from the training set, read in from my Rdata file containing training and testing set indicies.

3. Run `cv.glmnet()` which performs 10-fold cross-validation and outputs an intercept term and standardizes the variables by default. For ridge regression we use 'alpha =0' and for lasso we use 'alpha =1'.

4. `cv.glmnet()` will output a list of models. We decide the best one based of the minimum lambda and then save this lambda value as well as the coefficients associated with it.

5. Next we plot the model and save it to a png.

6. Once we identified the best model we use the **test_set** to calculate the test MSE, which will eventually help us compare the performances of all the models.

7. Finally we refit the model to the *scaled-client.csv* which is our entire data set using the lambda from step 4. We save the coefficient estimates and use it in the *Results* section of the report.

Our next two methods, Principal Components(PCR) and Partial Least Squares regression(PLSR) are dimension reduction methods performed by **PC_regression.R** and **PLS_regression.R** respectively. The steps for these two regression are very similar to those above, so naturall this outline will not go into as much detail.

1. Load *scaled-client.csv*, and `library(glmnet)` and set the seed, for reproducibility sake.

2. Create a `x` and `y` variable from the training set, read in from the Rdata file containing training and testing set indicies.

3. Run `pcr()` or `plsr()` depending on which model you want, and use arguments `Grad_Rate   ., data=train_set, validation = CV and scale = TRUE`.
4. We decide the best model using `which.min(MODEL`*validation*`PRESS)` where `MODEL` is the name of the pcr or plsr model depending on the script.
5. Next we plot the model and save it to a png.
6. Once we identified the best model we use the **test_set** to calculate the test MSE, which will eventually help us compare the performances of all the models.
7. Finally we refit the model to the *scaled-client.csv* which is our entire data set using the lambda from step 4. We save the coefficient estimates and use it in the *Results* section of the report.

6 Analysis

```
## Loading required package:  Matrix
## Loading required package:  foreach
## Loaded glmnet 2.0-5
##
## Attaching package:  'pls'
## The following object is masked from 'package:stats':
##
##    loadings
```

We perform Mean centering and standardization on our data in order to make sure that the coefficients would function properly and in an equivalent manner

6.1 Ordinary Least Squares Regression (OLS)

```
## [1] 0.2013151
```

We first fit the Ordinary Least Squares Regression model to the data set, with no parameters to be tuned, using hte ‘lm()‘ function. The mean square error of the OLS model on the test set is shown above: 0.201315062. We use this as a reference point when comparing with other prediction models.

6.2 Ridge Regression (RR)

We fit the Ridge Regression model to the training set. Firstly, we need to train the model for the best parameter λ via cross-validation (‘cv.glmet‘). Using ‘predict()‘ and calling the library package ‘ *glment* ‘ to perform the regression, we get the following:

```
## [1] 0.1232847
## [1] 0.2653839
```

From the graph above, we choose the best λ given this traning set, which is 0.123 and applied it to the entire data set

The resulting mean square error for the same test set is given above. The 0.265383914 is larger compared to that of the OLS Regression.

6.3 Lasso Regression (LR)

We fit the Lasso Regression model to the training set using the same method, but the difference is that the alpha variable in the function ‘cv.glmnet()‘ is changed from 0 to 1.

```
## [1] 0.01321941
## [1] 0.322184
```

From the graph above, we choose the best λ given this traning set, which is also 0.0132 and applied it to the entire data set.

The resulting mean square error for the same test set is given above. The 0.322 does show a slight deterioration compared to that of the OLS Regression.

6.4 Principal Components Regression (PCR)

We fit the Principal Components Regression to the training set by installing the library package ‘pls‘ and using the function ‘pcr()‘ to run the cross-validation using the opitimal lambda.

From the validation plot (displayed in the next section), we may find that the lowest cross-validation error occurs when $M = 11$ component are used. Therefore, we compute the test MSE:

```
## [1] 0.3119941
```

The resulting mean square error for the same test set is given above. The 0.311994108 is slightly higher than the the OLS Regression’s, but lower than Lasso. We also fit the model with the best choice of M, the number of components to the entire data set for future purpose.

6.5 Partial Least Squares Regression (PLSR)

We fit the Partial Least Squares Regression to the training set by installing the same library package ‘pcr‘ and using the function ‘plsr()‘ to run the cross-validation using the opitimal lambda or tuning parameter.

From the validation plot (displayed in the next section), we may find that the lowest cross-validation error occurs when $M = 4$ component are used. Therefore, we compute the test MSE:

```
## [1] 0.2797164
```

The resulting mean square error for the same test set is given above: 0.2797164397 This too is larger than the OLS Regression's. We also fit the model with the best choice of M, the number of components to the entire data set for future purpose.

Below is a table that summarizes the MSE values for each regression:

```
## \begin{table}[ht]
## \centering
## \caption{Test MSE Values for the Regression Techniques}
## \begin{tabular}{lr}
##   \hline
## Regression & MSE \\
##   \hline
## OLS & 0.2013151 \\
##   Ridge & 0.2653839 \\
##   Lasso & 0.3221840 \\
##   PCR & 0.3119941 \\
##   PLSR & 0.2797164 \\
##    \hline
## \end{tabular}
## \end{table}
```

So, apart from the OLS Regression's MSE, the Ridge regression's MSE is the smallest from the 4 other methods.

7 Results

In this section we study and compare the outputs for each regression.

7.1 Lasso and Ridge Regression

Below are the analogous plots for LR and RR. These plots compare log(lambda) values to MSEP.

The lambda values in LR and RR determine to what extent the Ridge and Lasso Regression models shrink the effect of regression coefficients. This shrinkage subsequently influences the MSEP errors as seen in the plots. We can note that lasso regression works best with very small lambda values (breaks in the plot) while ridge regression increases in error gradually (continuity). Since Lasso only uses a subset of the coefficient vector while Ridge does not, Lasso involves fewer predictive elements and so a larger tuning (lambda) values would more heavily influence the prediction, and thus the MSE.

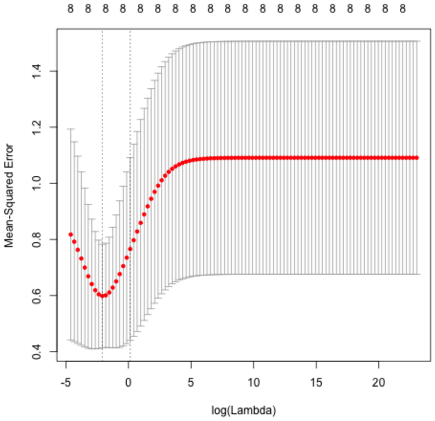

Figure 6: MSE Against lambda values for Ridge regression

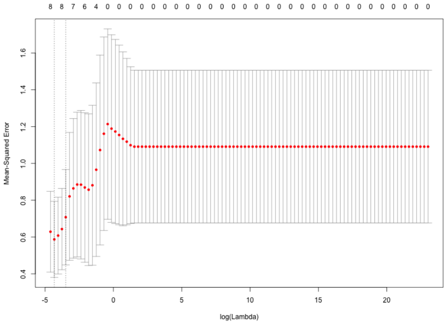

Figure 7: MSE Against lambda values for Lasso regression

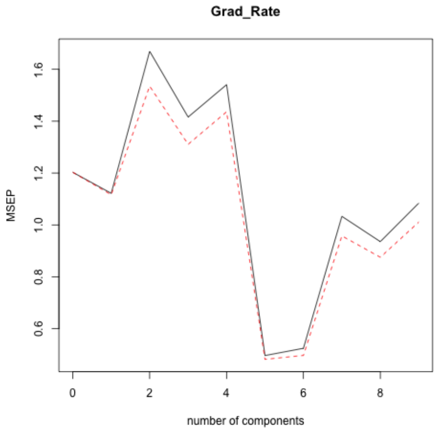

Figure 8: Validation Plot for PC Regression

7.2 PCR and PLSR

Below are the plots for cross-validation for PCR and PLSR. They compare the number of components used to the cross-validation MSE (MSEP).

How the dotted red line fits the solid black line indicates how accurate this regression model is for an arbitrary data set. From the graph, it appears as though PCR better matches up the expected error with the training set error since the lines are further apart for PLSR. This can be seen because PLSR minimizes error (MSEP) using fewer components than does PCR, and so there is a potential tradeoff in uncertainty, hence the gap between the lines is accentutaed for PLSR.

7.3 Additional Plots

```
## \begin{table}[ht]
## \centering
## \begin{tabular}{rrrrrr}
##   \hline
##  & OLS & Ridge & Lasso & PC & PLS \\
##   \hline
## Intercept & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\
##   Enroll\_White & -0.3564 & -0.2726 & -0.3308 & -0.2245 & -0.3016 \\
##   Enroll\_Black & 0.4494 & 0.3947 & 0.4332 & 0.4224 & 0.4346 \\
##   Enroll\_Hisp & -0.0545 & -0.0991 & -0.0498 & -0.2790 & -0.1276 \\
##   Enroll\_Asian & 0.2402 & 0.2408 & 0.2342 & 0.3061 & 0.2676 \\
##   Enroll\_Women & -0.0853 & -0.0768 & -0.0697 & -0.1230 & -0.1466 \\
##   Avg\_Age & -0.2724 & -0.2743 & -0.2907 & -0.2269 & -0.3083 \\
```

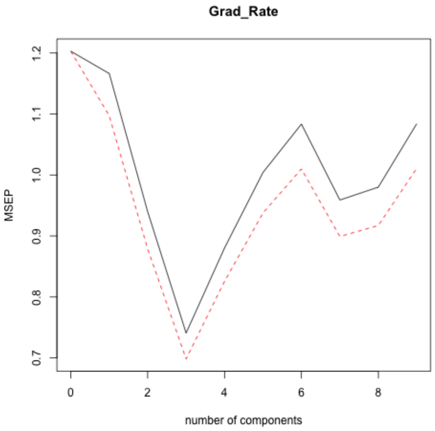

Figure 9: Validation Plot for PLS Regression

```
##   Avg\_Fam\_Inc & 0.3880 & 0.2661 & 0.3437 & 0.1307 & 0.2640 \\
##   Total\_enroll & -0.2006 & -0.1758 & -0.1880 & -0.1872 & -0.1903 \\
##    \hline
## \end{tabular}
## \caption{Regression Coefficients}
## \end{table}
```

```
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
```

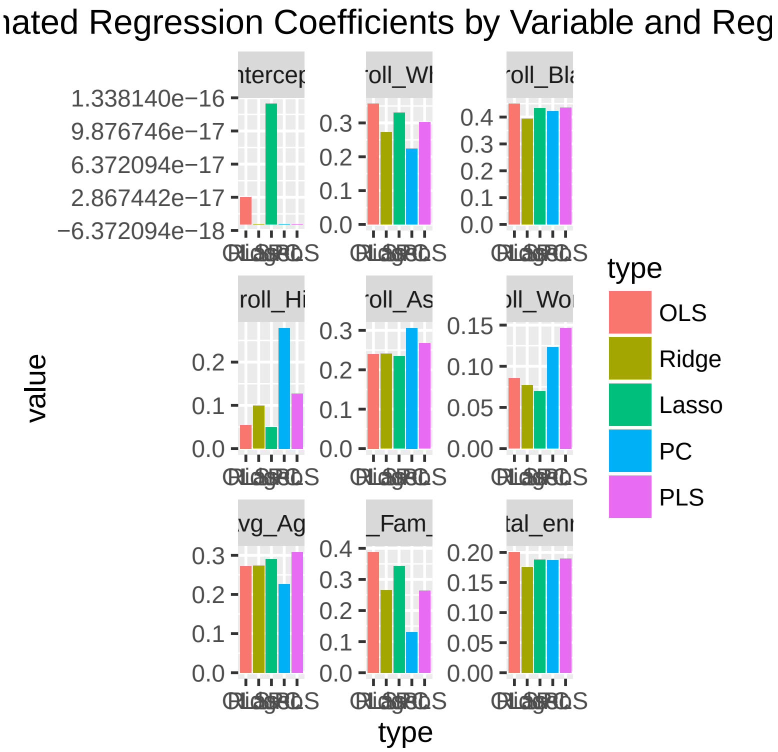

Figure 10: Estimated Regression Coefficients by Variable and Regression

```
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
## Warning in ‘levels<-‘(‘*tmp*‘, value = if (nl == nL) as.character(labels)
else paste0(labels, :  duplicated levels in factors are deprecated
```

8 Conclusion

In conclusion, through EDA analysis, we are able to provide a general overview of selected art institution in California regarding enrollment and graduation data. Through model building, we are able to show that there is a linear relationship between diversity variables and graduation rate in these art schools. Finally, we build an interactive shiny App for school administrators to visualize our analysis results

9 Shiny App

In order to gain more insight into our data set on California art schools, we made a shiny app that would facilitate users to understand the difference of certain statistics between each school and how they correlate to higher graduation rates. Using the shiny app, we are able to construct bar charts and pie charts that would give visual aid to readers on which school is the better of a certain pair. The following class types are the main indicators and predictors we used for our regression models: gender, ethnicity, graduation rate, average age, and median income. The selected groups consist of not only specific academic institutions, but also the three separated regions in California: Northern, Central, and Southern. Given an insufficient data set, there is no clean data for the art schools in Central California, so the charts are not shown.

One piece of information to consider is how graduation rates in Southern California are relatively higher than the rates in Northern California. Especially in graduation rates for the white race, the graduation rate in Southern in California is higher by approximately two-fold. From the ethnicity class type, we can see that Asians and Hispanics are enrolled more in Southern California. This case may be related to a regions demographic information, but it may be an indicator that having more Asians or Hispanics can lead to a higher graduation rate. In addition, comparing two schools from Southern and Northern California (Argosy University in San Diego and the San Francisco Conservatory of Music), we can see that the average age is relatively higher in San Diego while the median income is relatively lower in San Diego. A high average age with low median income can be related to lower graduation rates; however, a higher age group is more matured in terms of careers and academics, so a higher age group can be correlated to higher graduation rates.