Predicting Early Career Earnings After Graduation

STA3241.01 – April 27, 2020

Luiz Gustavo Fagundes Malpele, Cindy Nguyen, Isabel Zimmerman

Table of Contents

Introduction

The aim of this report was to predict early career pay for college graduates from data collected on independent variables such as: tuition costs, school enrollment size, percentage STEM of majors, percentage of minority students, etc. This is an important question for us, as we are college students who are soon entering the workforce. However, the outcomes of this study go beyond just students; this data could help forecast future pay and show the importance (or non-importance) of a college degree. After a robust exploratory data analysis, various regression algorithms were created utilizing the libraries tidyverse, caret, DataExplorer, fastDummies, leaps, cowplot, and GGally. All of the R code can be found at the GitHub here.

Dataset

We began by importing processed data from TidyTuesday, which can also be found here. From this data, we transformed all minority variables into percentages of total enrollment and took the log of the following variables: ***early career pay***, ***mid-career pay***, ***in state tuition***, ***out of state tuition***, ***room and board***, and ***total enrollment***. This was to create more normal distributions in the data and possibly remove heteroscedasticity. With the addition of these two features in the data, some of the models (particularly linear models) may improve in predictive power.

Data Dictionary

Table 1: Data Dictionary

Field Name	Description	Data Type
name	Institution Name	*factor*
state_code	State Abbreviation	*factor*
make_world_better_percent	Percent of alumni who think they are making the world a better place	*integer*
room_and_board	Room and board in USD	*integer*
ln_room_and_board	Natural Log of Room and board in U$D	*double*
early_career_pay	Estimated early career pay in USD	*int*
ln_early_career_pay	Natural log of estimated early career pay in USD	*double*
mid_career_pay	Estimated mid career pay in USD	*int*
ln_mid_career_pay	Natural log of estimated mid career pay in USD	*double*
total_enrollment	Total enrollment of students	*double*
ln_total_enrollment	Natural Log of Total enrollment of students	*double*
out_of_state_tuition	Tuition for out-of-state residents in USD	*integer*
ln_out_of_state_tuition	Natural Log of Tuition for out-of-state residents in USD	*double*
in_of_state_tuition	Tuition for in-of-state residents in USD	*integer*
ln_in_of_state_tuition	Natural Log of Tuition for in-of-state residents in USD	*double*
stem_percent	Percent of student body in STEM	*double*
private	Type: 0 for Public, 1 for Private	*integer*
asian_ratio	Percentage of Asian Students	*double*
black_ratio	Percentage of Black Students	*double*
minority_ratio	Percentage of all Minorities Combined	*double*
hispanic_ratio	Percentage of Hispanic Students	*double*
women_ratio	Percentage of Women Students	*double*
tuition_ratio	Out-of-State Tuition and In-State Tuition Ratio	*double*

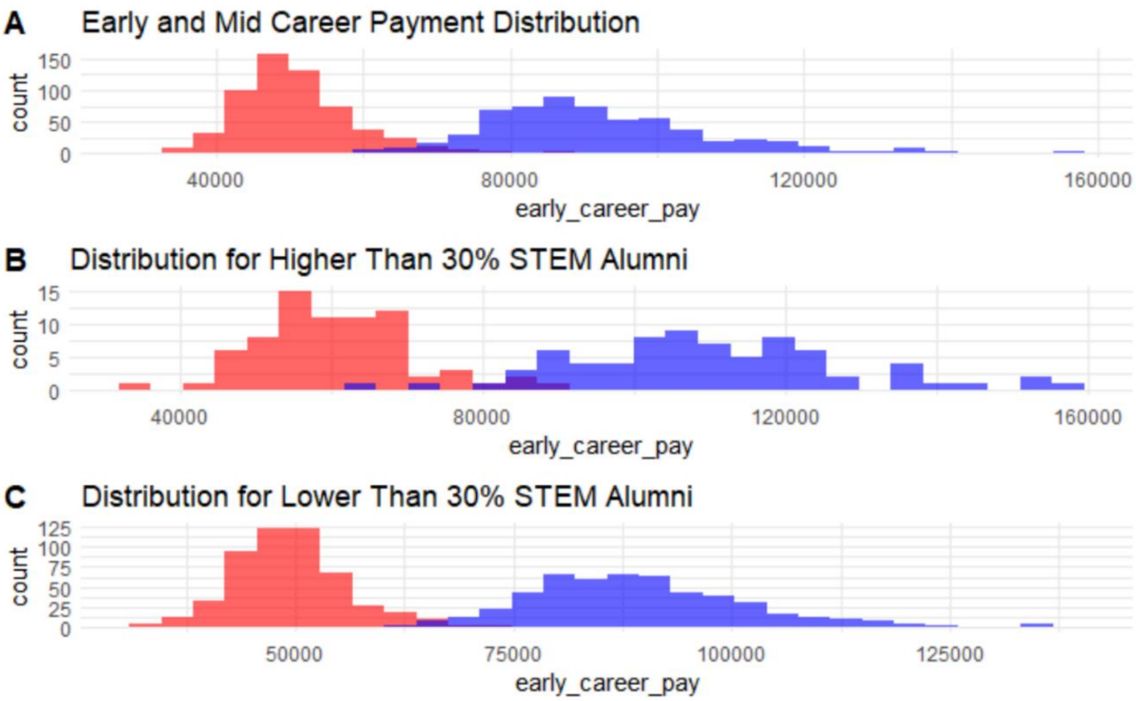
Exploratory Data Analysis

The first step was to use the DataExplorer package to automatically create an EDA. This report can be found here. Using this process was preferred as it automatically created all the univariate distributions and correlation matrices for the variables. This way, we were able to focus on creating more complex explorations that were fine-tuned to the question we wanted to answer.

The first look into the data was to see how the distribution of pay shifted from early to mid-career. We could tell that the distribution became wider and right-skewed for mid-career pay and was higher on average; the mean early pay was $51,000 whereas the mid-career pay average was $92,000.

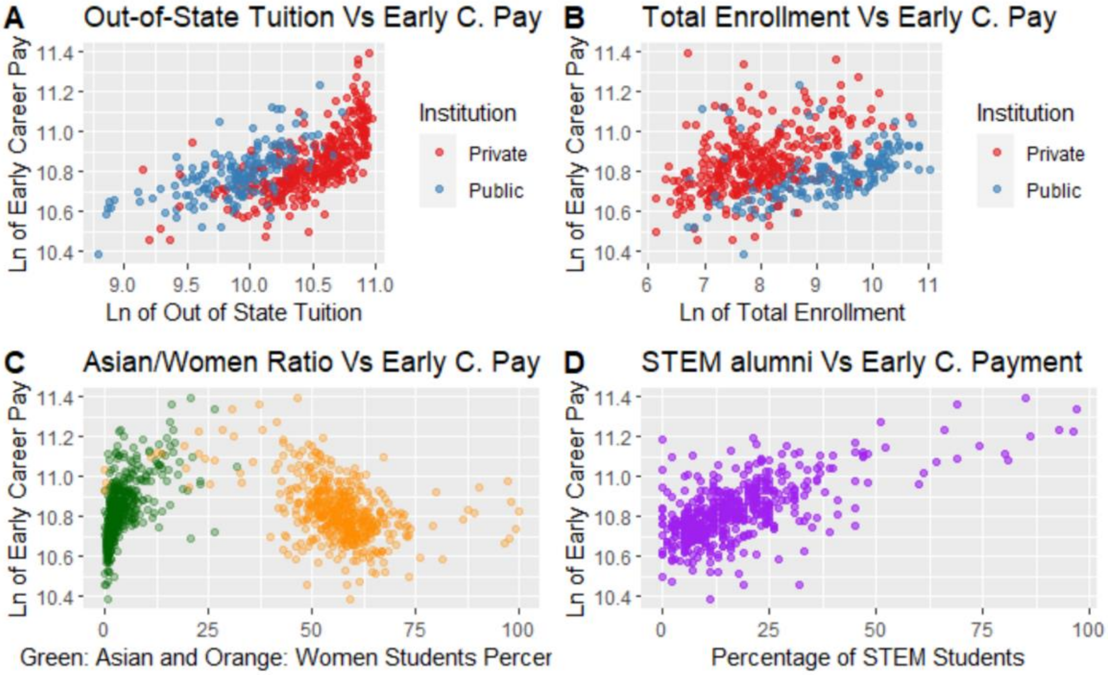
Then, we explored more thoroughly the impact of variables we thought would be highly significant in our regression models. The first variable we chose to explore was ***stem_percent*** as STEM majors tend to have higher paid jobs both right out of college and over time. In Figure A, it is observed that both early and mid-career pay has relatively normal distributions. However, when observing schools with higher than 30% STEM majors in Figure B, there is no longer a normal distribution; both early and mid-career pay are observed to be proportionally high but do note that the sample size is much smaller for this visualization. Finally, in Figure C, we see that the less than 30% STEM majors have a relatively similar distribution as the school totals; that is, this distribution is approximately normal.

Figure 1: Variable Distribution of Early and Mid-Career Payments



After seeing the impact of just STEM majors, we decided to look at other indicators in comparison to our response variable of ***early_career_pay***. We decided to use scatter plots due to their ease of interpretability and ability to see each individual datapoint as well as the patterns of these relationships. In plots A, B, C, and D below, you can see how out of state tuition, total enrollment, minority groups, and STEM majors change with early career pay. These variables were chosen as they consistently proved to be statistically significant in the EDA created by DataExplorer and in our further exploration that can be seen on the GitHub repo.

Figure 2: Scatter Plots of Explanatory Variables against natural log of Early Career Payment



Plot A shows that the higher the out of state tuition, the higher will be the early career payment for a student. The social background behind this relationship shows that a student who attends a more expensive college tends to make more money in the future and that this student most likely comes from a wealthier family. Furthermore, this scatter plot captures a clear division between Public and Private universities since the private ones are clearly more expensive.

Plot B present a positive relationship between Total Enrolment and Career Payment. In most instances of that scatter plot, the higher the total enrollment, the higher the career pay, especially for private universities.

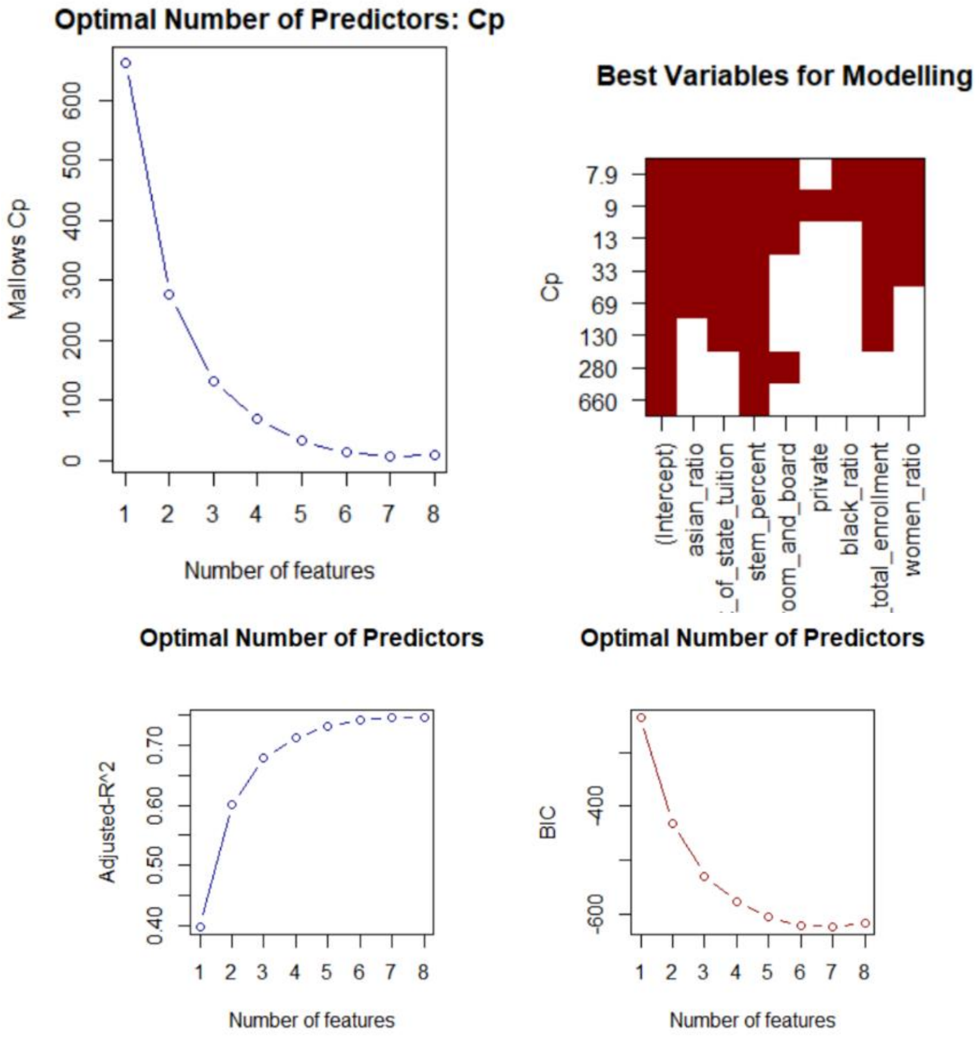
Plot C presents data about gender (women percentage) and a specific ethnicity (percentage of Asian students) against early career pays. The negative relationship between women ratio and early pays has a historical background and according the Institute of Women's Policy Research "In 2018, female full-time, year-round workers made only 0.82 cents for every dollar earned by men, a gender wage gap of 18 percent."[1] Lastly, asian_ratio had a positive relation to early payments.

Plot D shows a direct relationship between Early Career Payment and the percentage of Students within the institution, the higher the STEM percentage, the higher will be the estimated early career payment.

Modeling

Variable Selection

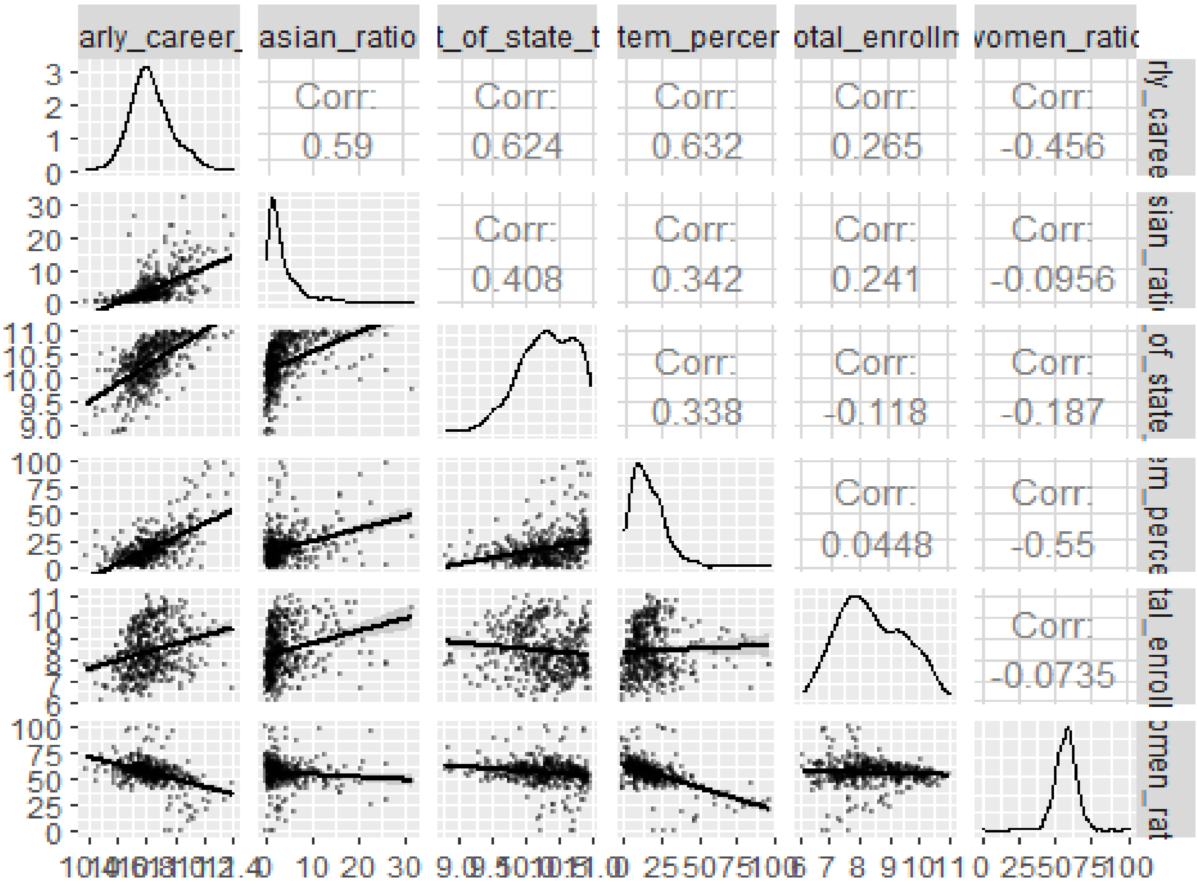
Figure 3: Variables Selection Plots



[1] Quote sourced from: https://iwpr.org/issue/employment-education-economic-change/pay-equity-discrimination/

Based on the EDA, BIC, Mallows' CP, and the $Adjusted - R^2$ in the Figures above, the models will be tested on the following predictors: ***ln_early_career_pay, asian_ratio, ln_out_of_state_tuition, stem_percent, ln_total_enrollment***, and ***women_ratio***. If we use any more predictors, this will result in overfitting.

Figure 4: Correlation Matrix and Bivariate Plots



We then formed a correlation matrix between these variables; if variables are highly correlated, they can cause standard errors of models to be unreliable and cause poor models in general. From this, we found that the variables were, at most, about 60% correlated. This was not worrisome, but certainly something to keep in mind when evaluating results.

For testing purposes, we created a train control variable in order to establish that each model would be tested with 10-fold cross-validation. This is to ensure that the models are not overfitting in the training phase, and it gives feedback on how well the model is performing. We also split the data so that 80% of arbitrary but specific data is used to train, and the other 20% is used to test the model's performance. This is also done to avoid overfitting, and it is preferable to perform the final model selection with an out of sample criterion.

Best Model

Random Forest

Random Forest happens to be one of the most popular algorithms in data science as it can both classify and regress data. As in the name, a random forest is made of n number of individual decision trees that work together to provide accurate results. This is helpful in our project; some models that we are predicting may be inaccurate while some may have better results. Having a plethora of results that forms informational analysis will help with having less error in our data. Furthermore, the random forest model can be displayed as a decision tree, and it is easy to interpret by people out of the Data Science field since it mirrors the human decision-making process.

The random forest utilized 391 samples and 6 predictors. The best model had the smallest RMSE was of 0.07268202 and used 2 decision trees.

```r
oob <- trainControl(method = "oob")
cv_5 <- trainControl(method = "cv", number = 5)
rf_grid <- expand.grid(mtry = 1:10)

rf_model <- train(ln_early_career_pay ~ ., data = train_data,
                     method = "rf",
                     trControl = oob,
                     verbose = FALSE,
                     tuneGrid = rf_grid)
```

Figure 5: Comparison of Actual and Forecast results for Train and Test Data

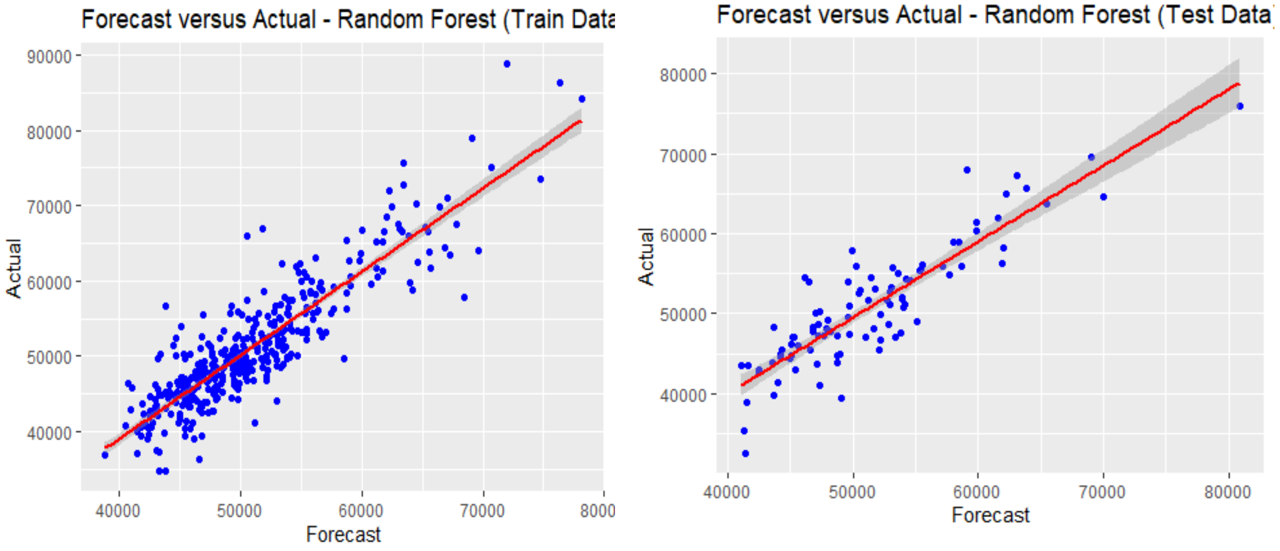
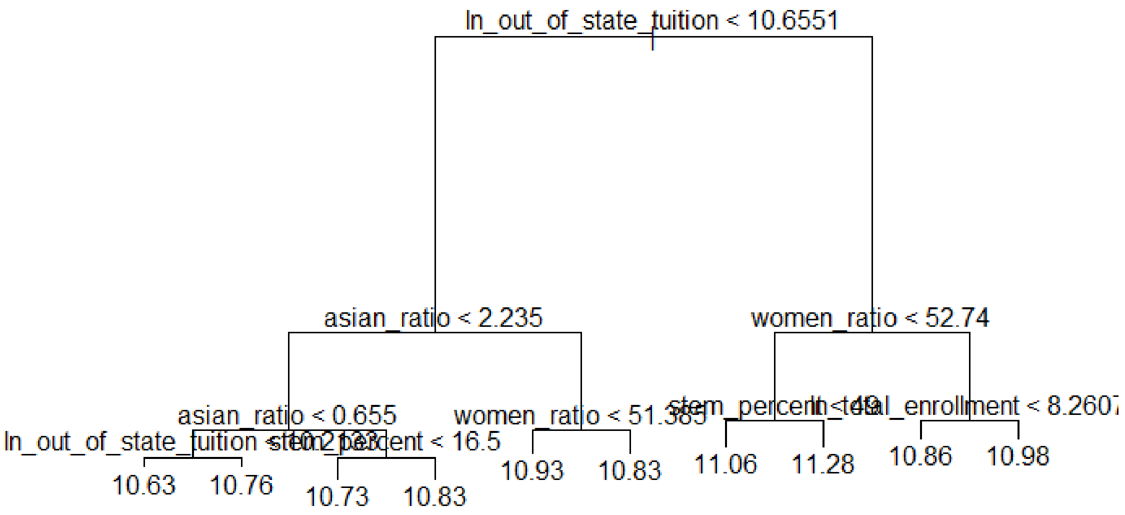


Figure 6: Random Forest Final Tree Graphically represented



Model Comparison

For brevity, the model selection results are posted below. We chose to highlight the random forest model as it has the highest R^2 and the lowest in and out of sample RMSE. In order to see our analysis of the other models, see below in the *Other Techniques* section.

Table 2: Model Selection Criteria

Predictive Model	R-Squared	In Sample RMSE	Out of Sample RMSE
Ordinary Least Squares	0.7382	0.07665	0.08076483
Ordinary Least Squares-glmnet	0.07715067	0.7468707	0.08027886
Random Forest	0.7640539	0.07268202	0.0727503
Principal Component Analysis	0.7417742	0.07682088	0.08076483
Support Vector Machine	0.7367368	0.07692738	0.07923666

Other Techniques

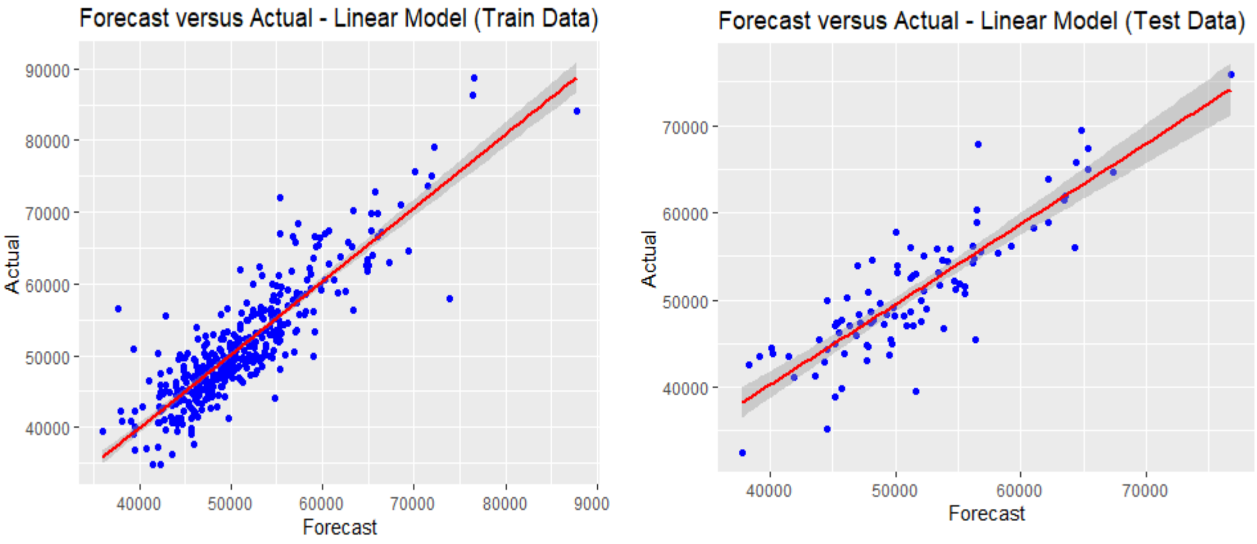
Ordinary Least Squares

This Ordinary Least Squares linear model is focused on the variable, ***ln_early_career_pay***, and is being tested with seven other variables that were previously selected by the previous methods. The Adjusted R^2 is 0.7382 and all predictors are statistically significant to the analysis.

```
earlypay_lm <- lm(ln_early_career_pay ~ .,
                  data = train_data)


## Call:
## lm(formula = ln_early_career_pay ~ ., data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.24361 -0.04826 -0.00374  0.04307  0.40764
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)               9.1334438  0.1200992  76.049  < 2e-16 ***
## asian_ratio               0.0091614  0.0010750   8.522 3.60e-16 ***
## ln_out_of_state_tuition   0.1425788  0.0102636  13.892  < 2e-16 ***
## stem_percent              0.0029324  0.0003481   8.423 7.37e-16 ***
## ln_total_enrollment       0.0308594  0.0038493   8.017 1.31e-14 ***
## women_ratio              -0.0020555  0.0003963  -5.187 3.46e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07665 on 385 degrees of freedom
## Multiple R-squared:  0.7416, Adjusted R-squared:  0.7382
## F-statistic:   221 on 5 and 385 DF,  p-value: < 2.2e-16
```

The Root Mean Square Error of the out-of-sample prediction was calculated by utilizing the testing set of the mean of the following difference squared: $(\hat{y} - y)^2$, also known as RMSE, the result was 0.07665. This is another linear graph that shows a comparison of the Actual and Forecast values, but only the test set or 20% of the data was used. Again, the data is mostly surrounded around 50,000.

Figure 7: Comparison of Actual and Forecast results for Train and Test Data

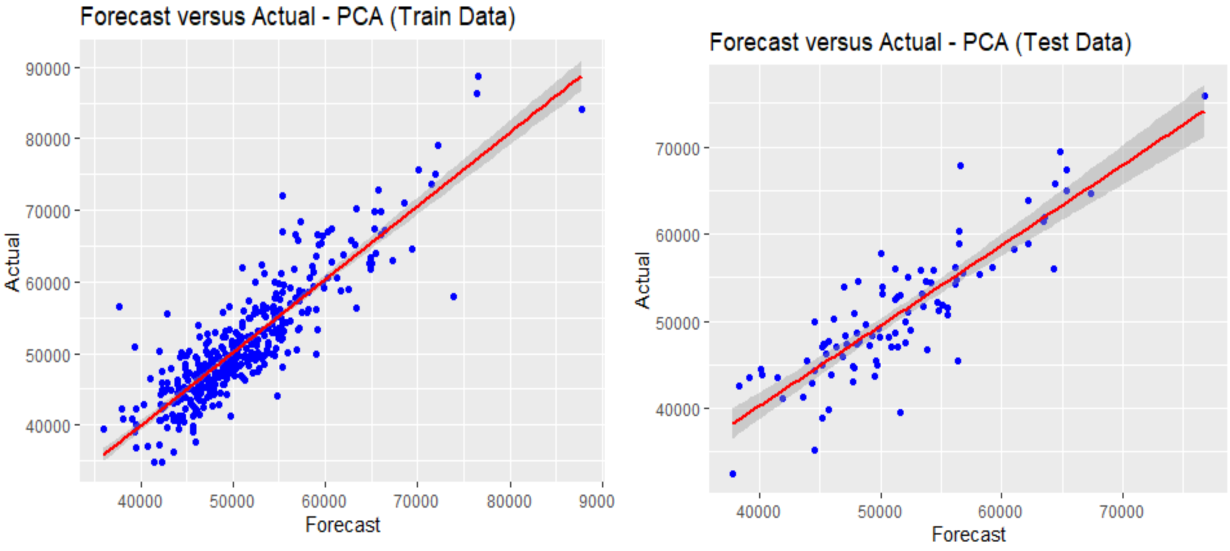


Principal Component Analysis

```
glm_pca_model <- train(ln_early_career_pay ~ . ,
                 data = train_data,
                 method = "glm",
                 preProcess = "pca",
                 trControl = train_control)
## Generalized Linear Model


## Pre-processing: principal component signal extraction (5), centered (5),
##  scaled (5)
## Resampling: Cross-Validated (10 fold)
## Resampling results:
##
##   RMSE        Rsquared   MAE
##   0.07753735  0.7365653  0.0580205
```

Principal Component Analysis, or PCA, is a type of linear transformation that allows you to visualize the overall format of the dataset. In a way, PCA “tilts” the dataset to be one dimensional. This will depend on the number of variables and will help to understand what variables are like each other and which are different. We utilized PCA to reduce the dimensionality of our dataset to make it easier to work with. In the linear model above, we have 391 samples with 8 predictors. The R-squared value of 0.7365653 tells us that the model that we are running is fitting the actual data by 73.6%. It is ideal for *RMSE* values to be as small as possible, or as close to zero on a zero to one scale. The in-sample *RMSE* is 0.07682088.

Figure 8: Comparison of Actual and Forecast results for Train and Test Data

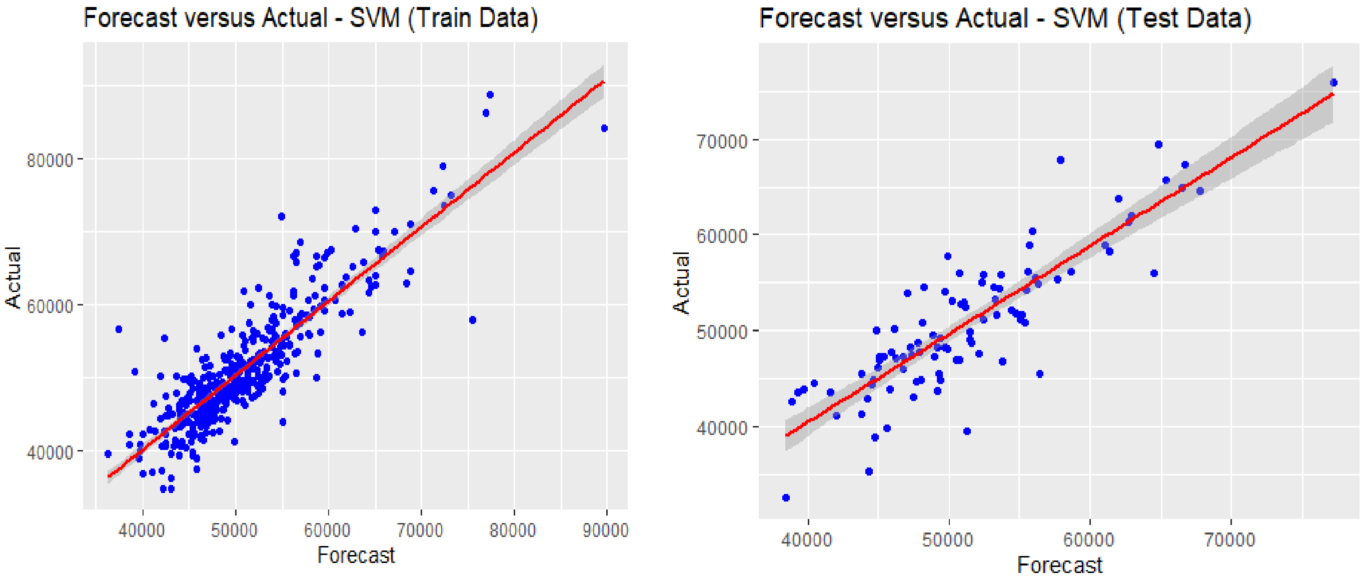


Support Vector Machine

```
svm_model_1 <- train(ln_early_career_pay ~ .,
  data = train_data,
  method = "svmLinear",
  tuneGrid = tGrid,
  trControl = tr_control,
  metric = "RMSE",
  preProcess = c("center", "scale")
)
## Support Vector Machines with Linear Kernel

##
##   C     RMSE        Rsquared   MAE
##   0.01  0.07774523  0.7364486  0.05804287
##   0.05  0.07692738  0.7367368  0.05746174
##   0.10  0.07698578  0.7367897  0.05749150
##   0.25  0.07695550  0.7367911  0.05742800
##   0.50  0.07699114  0.7366774  0.05745745
##   0.75  0.07701765  0.7364923  0.05746669
##   1.00  0.07698763  0.7366353  0.05746556
##   1.25  0.07704349  0.7365145  0.05749835
##   1.50  0.07705455  0.7364748  0.05751070
##   1.75  0.07705137  0.7364952  0.05750378
##   2.00  0.07705310  0.7365205  0.05751559
##   5.00  0.07707319  0.7363771  0.05752366
```

Support vector machine, SVM, is another type of classification and a regression model called Support Vector Regression, SVR, which we will be focusing on. Utilizing SVR will help us to minimize the sum of squared errors. It will also provide flexible analysis on how much room we can allocate for error and find a line that will fit the data points. If it is not on a linear boundary, hyperplane and multiple dimensions can be used to group data points together to produce the best and accurate values. We tuned the model c's parameter, which penalizes misclassification; our desirable c parameter was 0.05. The best model had an in sample RMSE of 0.07692738 and R^2 of 0.7367368.

Figure 9: Comparison of Actual and Forecast results for Train and Test Data



This SVM model shows 20% of the tested data on the dataset. It seems more distributed along the regression line with some outliers. The residuals are a little bit greater than the test model from random forest, whose data points were much closer to the regression line. Between the two models, we could say that random forests seem to provide a more accurate result.

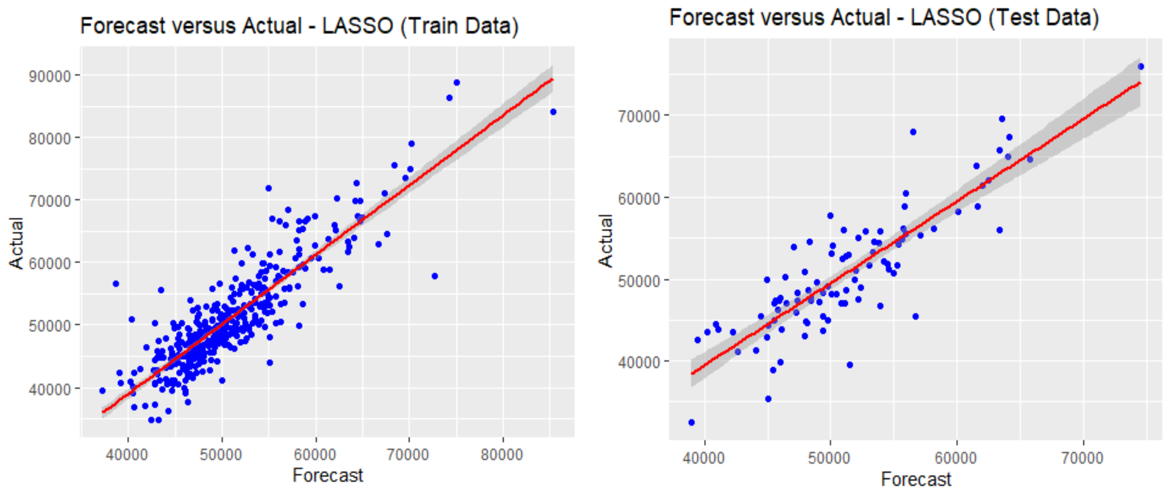
Ordinary Least Squares with Glmnet (LASSO/Ridge)

```
lasso_model <- train(ln_early_career_pay ~ .,
                     data = train_data,
                     method = "glmnet",
                     trControl = train_control,
                     metric =  "Rsquared",
                     tune_Grid = expand.grid(alpha = 1, lambda = grid))

## Resampling results across tuning parameters:
##
##   alpha  lambda        RMSE        Rsquared   MAE
##   0.10   0.0001917911  0.07651221  0.7463793  0.05778304
##   0.10   0.0019179111  0.07650888  0.7464208  0.05779062
##   0.10   0.0191791107  0.07715067  0.7468707  0.05841034
##   0.55   0.0001917911  0.07652939  0.7462609  0.05779879
##   0.55   0.0019179111  0.07657131  0.7461840  0.05786946
##   0.55   0.0191791107  0.07956271  0.7420766  0.06032027
##   1.00   0.0001917911  0.07653376  0.7462049  0.05780614
##   1.00   0.0019179111  0.07665041  0.7459095  0.05795561
##   1.00   0.0191791107  0.08373134  0.7296577  0.06410784


## The final values used for the model were alpha = 0.1 and lambda = 0.019179
11.
```

This **Ordinary Least Squares with LASSO penalization** linear model contains the seven variables previously used in the linear model tested against the ***ln_early_career_pay***. The best LASSO model has a $alpha = 0.1$ and $lambda = 0.01917911$. The highest R^2 is 0.7468707. LASSO increases the variance explained for the predictive model, but it also has a small penalty increasing the bias.

When the test set was used for an out of sample prediction, the regression line for the Forecast versus Actual values presents a better result when compared to the simple OLS model. Observation fall closer to the line and the Out of Sample RMSE is 0.62077, which does not represent a significant increase in bias, when compared to gain on explanatory power when the LASSO penalization was used.

Figure 10: Comparison of Actual and Forecast results for Train and Test Data



Conclusion

In the end, our Random Forest algorithm had the smallest RMSE of 0.07268202; this model outperformed SVMS, linear models, principal component analysis, and ordinary least squares regression with LASSO penalization. It was not particularly surprising that the random forest model was the highest performing, as random forests are ensemble techniques that aggregate the results of multiple decision trees to create a more stable estimation. This model could help new colleges and universities such as Florida Poly can input information about their students and receive estimations of early career pay to use for marketing or recruiting techniques. If we wanted to continue our research, we could use the model for early career pay and apply it to mid-career pay in order to understand what types of universities, degrees, and groups of people show the most growth in pay throughout their career.