Estimating Earnings After Graduation from Post-Secondary Institutions

Alex Van Rooy

2023-04-06

Table of contents

Index **6**
Terms and Abbreviations 6

Abstract **7**

1 Introduction to Project **8**
1.1 Overview 8
1.2 Objectives of Project 8

2 Pre-Processing and Exploratory Data Analysis **9**
2.1 Dataset Collection 9
2.2 Data Pre-processing 9
2.2.1 Converting Columns to Numeric 10
2.2.2 Dealing With Missing Values 10
2.2.3 Scaling Data 10
2.3 Feature Selection 10

3 Methodology **13**
3.1 Introduction to Python for Machine Learning 13
3.2 Data Split 13
3.3 Model Planning 13
3.3.1 SVM 14
3.3.2 Decision Tree 14
3.3.3 KNN 14
3.3.4 Lasso 14
3.4 Model Training 15
3.4.1 Training SVM 15
3.4.2 Training Decision Tree 16
3.4.3 Training KNN 16
3.4.4 Training Lasso 16
3.5 Final Model Building 17

4 Results **18**
4.1 Performance Metrics 18
4.1.1 R-Squared 18
4.1.2 RMSE 18
4.1.3 MAE 19
4.2 Results Table 19

4.3 Interpretation of the Results 19
4.4 Visualization 20
4.4.1 Interesting Correlations 21

5 Conclusion **24**

References **25**

List of Figures

4.1 Accuracy of Models Within Different Intervals 20
4.2 Model Fit 22
4.3 Correlations 23

List of Tables

2.1 Features Used 11
2.2 Basic Feature Description 12

4.1 Results from Models 19

Index

Terms and Abbreviations

SVM - Support Vector Machine

KNN - K-Nearest Neighbor

MSE - Mean Square Error

RMSE - Root Mean Square Error

MAE - Mean Absolute Error

Abstract

Deciding on which post-secondary institution to attend can be a very daunting task. For many students their choice of institution will be the first step in reaching their future career goals, and the hope for any student is that the institution they attend will help them reach success in their chosen field. One major metric of success is how much they will earn after leaving school, and for students it can be a very strong driving force when deciding between institutions.

To assist students in finding the institution that meets their financial needs/goals, we created a Machine Learning model to predict earnings post-graduation using various characteristics of an institution. Data was gathered from the College Scorecard website and it was used to train 4 different supervised machine learning regression models: SVM, Decision Tree, KNN, and Lasso.

The model that proved most accurate was the SVM model. It was able to achieve an R-Squared of around 0.8 and predict the earnings post-graduation with an RMSE of 4527. In addition to creating a model that could predict future earnings, interesting correlations between some of the features used to train the model were found. These correlations were: student completion rate vs school's net tuition revenue, family income vs future earnings, and withdrawal rate vs median family income.

This model serves to assist not just students but institutions as well in understanding what features may play a role in the future earnings of their graduates. Through use of this model the hope is that students gain deeper insight into the institutions they wish to attend, as well as assist them in selecting a school that meets their financial goals.

1 Introduction to Project

The task of picking an institution to complete a post-secondary education at is a very intimidating one. More and more information is being made available to students so that they can guide their decision with information relevant to them and their future goals. One thing that many students consider when deciding on which school to attend, is how successful they will be after graduating from an institution. Knowing what a student can expect after graduating from an institution can be a great benefit to them when picking between schools. Although success is measured in many ways, one metric of success is earnings. By having knowledge about how much a student may earn after graduating from an institution, they can tailor their selection of potential schools based off their future goals and the trade-off between costs to attend a school versus what they may get after graduating.

To solve this problem, this report proposes a machine learning learning model that will take various characteristics of an institution and predict the future earnings of a student, 6 years after graduation.

1.1 Overview

This report is split into sections, they are: Pre-Processing and Data Exploration which will explain the dataset that is used and how it was prepared, Methodology which explains the models that were selected and how they were trained to the data, Results which reports and analyses the outcome after testing the trained models, and Finally Conclusion which summarizes the findings and discussed potential future work.

1.2 Objectives of Project

This project has two objectives. The primary objective is to create a model that uses various characteristics of a post-secondary institution and produces an accurate and meaningful estimate of what the future earnings a student graduating from a specific institution may look like. The secondary objective is to provide institution and students with some deeper knowledge about what factors may contribute to more earnings post-graduation from specific institutions.

2 Pre-Processing and Exploratory Data Analysis

Before any of the models were trained, tested and compared, the raw dataset was first processed to make it suitable for use. The sections below outline the data collection as well as data pre-processing steps that were carried out in order to produce a dataset suitable for use.

2.1 Dataset Collection

The dataset that was used for this program is the U.S. Department of Education's College Scorecard Dataset, which is publicly available for download here. They offer two datasets, one pertains to information by Field of Study, the other pertains to Institution-Level Data. For the purposes of this project we used the Institution-Level Dataset.

The College Scorecard Project is designed for the purpose of providing future students with information about the potential costs and outcomes of different colleges. It's goal is to provide transparency with families so they can look through the data and make judgments as to meet their own financial and educational goals. They collected their information through federal reporting from institutions, data on federal financial aid, and tax information.

The information in the dataset relates to topics such as: costs, student body demographics, admissions, earnings, and debt, just to name a few. Most importantly, for each institution the dataset had information on the mean earnings for their graduates, which will be used as the target variable in model training and prediction.

The dataset directly from the website cannot be used to train the models so it first must be cleaned and processed, which will be discussed in the next section.

2.2 Data Pre-processing

The raw dataset had 6681 rows and 2989 columns before it was cleaned and processed. In other words, there are almost 3000 features per institution in the dataset. This amount of features is much too large for the scale of this project, but also, with these many features it could affect the model performance and it would be hard to generate any meaningful analysis with so many variables to look through. Thus through pre-processing, the number of features used to train the models is cut down significantly.

2.2.1 Converting Columns to Numeric

The first problem with the dataset is that some of the columns are of mixed datatype, meaning some elements of a column may be numeric while others are text based. One reason why this is the case is because for some of the entries in the dataset, the value "PrivacySupressed" was given to them in order to protect the privacy of individuals, and this can lead to a column having a mix of numeric values and "PrivacySupressed" values.

To deal with this, all columns that were not fully numeric were gathered and the elements in the columns were converted to numeric. Elements that did not have a numeric representation were treated as NaN, i.e. missing. The majority of the dataset's features were numeric so this did not affect any important features.

2.2.2 Dealing With Missing Values

When the dataset is first examined, about 18% of its values are NaN. After performing the procedure explained in the previous section, that number rises to 63% of the values are NaN. To deal with these missing values, each column and row is examined. If a column has over 40% of it's entries as NaN then it is considered a 'bad column' and it is dropped. Similarly if a row has over 40% of its values as NaN then it is considered a 'bad row' and it's removed. The purpose of using this threshold approach is that it allows us to remove columns/rows that offer very little real data, while also preserving columns/rows that have missing data but have a majority real data. After this procedure is performed the number of NaN values drastically drops to around 9% and the dimensions of the dataset change to 5003 rows and 732 columns.

With the remaining 9% of missing values, we replaced each of them with the mean value of the column they appeared in.

Now the dataset has no missing values and is ready for further processing to make it ready for training models.

2.2.3 Scaling Data

The values in the dataset range in magnitude and thus it may be appropriate to scale the data. However, scaling/standardizing the data does not necessarily improve the performance of all models so the scaling of data is handled at the model level right before training the model. The models in this project that utilized scaling/standardizing are SVM, KNN, and Lasso. For more information regarding scaling please refer to Chapter 3.

2.3 Feature Selection

Before the dataset can be used for training the models, the features and target need to be extracted.

For the target variable the feature "MN_EARN_WNE_P6" will be used. This variable represents the mean earnings of a student 6 years after they graduated.

The dataset has 732 columns that can be used in the feature selection process. Due to the quantity of possible features and limited processing power, it was too computationally intensive to find the best features using feature extraction techniques. Instead the features were hand selected based on relevance to the end goal. It is also important to note that although there were 732 features, many of the features were subdivisions of a larger category, and thus they reported very similar data. The number of features selected was 82, they are displayed in the table below.

Table 2.1: Features Used

SCH_DEG	PPTUG_EF	FTFTPCTFLOAN
MAIN	TUITFTE	CNTOVER150_1YR
NUMBRANCH	INEXPFTE	
PREDDEG	POOLYRS	
HIGHDEG	POOLYRS200	
CONTROL	PCTFLOAN	
CCBASIC	CDR2	
CCUGPROF	CDR3	
CCSIZSET	COMP_ORIG_YR2_RT	
DISTANCEONLY	WDRAW_ORIG_YR2_RT	
UGDS	COMP_ORIG_YR3_RT	
UGDS_WHITE	WDRAW_ORIG_YR3_RT	
UGDS_BLACK	COMP_ORIG_YR4_RT	
UGDS_HISP	WDRAW_ORIG_YR4_RT	
UGDS_ASIAN	COMP_ORIG_YR6_RT	
UGDS_AIAN	WDRAW_ORIG_YR6_RT	
UGDS_NHPI	COMP_ORIG_YR8_RT	
UGDS_2MOR	WDRAW_ORIG_YR8_RT	
UGDS_NRA	RPY_1YR_RT	
UGDS_UNKN	COMPL_RPY_1YR_RT	
NONCOM_RPY_1YR_RT	FEMALE_DEBT_N	
RPY_3YR_RT	MALE_DEBT_N	
COMPL_RPY_3YR_RT	FAMINC	
NONCOM_RPY_3YR_RT	MD_FAMINC	
RPY_5YR_RT	FAMINC_IND	
COMPL_RPY_5YR_RT	PCT_WHITE	
NONCOM_RPY_5YR_RT	PCT_BLACK	
RPY_7YR_RT	PCT_ASIAN	
INC_PCT_LO	PCT_HISPANIC	
INC_PCT_M1	PCT_BA	
INC_PCT_M2	PCT_GRAD_PROF	
PAR_ED_PCT_PS	PCT_BORN_US	
DEBT_MDN	MEDIAN_HH_INC	
GRAD_DEBT_MDN	POVERTY_RATE	
WDRAW_DEBT_MDN	UNEMP_RATE	
FEMALE_DEBT_MDN	ICLEVEL	

MALE_DEBT_MDN	UGDS_MEN
DEBT_N	UGDS_WOMEN
GRAD_DEBT_N	OPENADMP
WDRAW_DEBT_N	FTFTPCTPELL

For a detailed explanation please refer to the College Scorecard website.

The basic description of these features is displayed in Table 2.2.

Table 2.2: Basic Feature Description

Feature Name	Description
SCH_DEG	Institution Award
MAIN	Is this the main campus?
NUMBRANCH	How many campus' does this institution have
PREDDEG	Identifies the type of award that the institution primarily confers
HIGHDEG	Highest award level conferred at the institution
CONTROL	Identifies what type of institution this is (public, private, etc.)
CCBASIC	Basic Carnegie Foundation Classification
CCUGPROF	Carnegie Foundation Undergraduate profile
CCSIZSET	Carnegie Foundation Size and Setting Classification
DISTANCEONLY	Is the institution online only?
UGDS_*	Number of undergraduates
NONCOM_RPY_*	The debt repayment rate of students who withdrew before completion
COMPL_RPY_*	The debt repayment rate of students who completed
RPY_	Repayment Rate
INC_PCT_*	Family Income
PAR_ED_PCT_PS	Share of students who's parents completed post-secondary education.
DEBT_*	Debt for differnt metrics
PPTUG_EF	Proportion of full-time/part-time undergraduates
TUITFTE	Net Tuition Revenue
INEXPFTE	Instructional Expenditures Per Full-Time Student
COMP_ORIG_*	Rate of students who completed studies at original institution
WDRAW_ORIG_*	Rate of students who withdrew from studies at original institution
FAMINC	Family Income

3 Methodology

The machine learning models used in this project had to be trained to the data before they could be used to make any meaningful predictions. Each model followed a similar training and optimization procedure which is outline in the following sections.

3.1 Introduction to Python for Machine Learning

Python is the programming language of choice for this project. Python has lots of machine learning support making it very easy to build and test models such as the ones presented in this report. The Scikit-learn open source machine learning library was used to build, train, and test all the models in this project (Pedregosa et al. (2011)).

3.2 Data Split

The first step in training a model is to have data which it will use to train and data that it will use to test the models performance. To achieve this, the cleaned dataset was split into two parts. The first part is known as the training set which will be the data that is used to train and fit the model. The second part is the test set which is used to evaluate the trained models on new data that was not seen before.

It is best to train the data on as many samples as possible, but also it is important to test the models on a large variety of samples to get an accurate representation of how the model performs. The split used was a 70/30 split which means that 70% of the dataset is used for training while the remaining 30% is used for testing. The pre-processed dataset had 5003 samples, after the split the training set had 3502 samples and the test set had 1501 samples.

3.3 Model Planning

The problem of predicting future earnings is a regression problem, therefore the models that will be used must work well with regression. Also, given that the dataset has many features it is highly unlikely that a linear model will be able to accurately capture the fit of the data, thus non-linear models must be considered when selecting a model. Finally, since the earnings of each sample in the dataset is know, the models will be supervised learning models.

With these requirements in mind, the models that were chosen are: Support Vector Machine (SVM), Decision Tree, K-Nearest Neighbor (KNN), and Lasso Regression. A brief discussion of the models and why they were selected are in the following sub-sections.

3.3.1 SVM

SVM's are complex supervised learning methods that work well in high dimensional spaces. Typically they are used in classification problems, however they still can be used for regression. Since the dataset being used is of higher dimension it was necessary to use a model that can work well with such data and SVM is one such model. SVM is a very complex model compared to other models, but it is also able to provide good results for the type multi-variate model that will be created ("1.4. Support Vector Machines" (n.d.)).

3.3.2 Decision Tree

Decision Tree models are used both in classification and regression. The learn simple decision rules from the data features and use these rules to make predictions. These decision trees can get very complex depending on how deep it is allowed to grow. The benefit of a decision tree is its interpretability, they can easily be visualized which makes it good for dealing with data that is hard to visualize such as the dataset this project is using. ("1.10. Decision Trees" (n.d.))

3.3.3 KNN

KNN is a very simple model conceptually, but it can provide great results. The KNN model works by taking a predefined number of training samples that are closest to the new point, and assigning a value to the new point based on those nearest points. KNN works for both classification problems and regression problems. It's simplicity makes it appealing when dealing with large datasets with lots of features. ("1.6. Nearest Neighbors" (n.d.))

3.3.4 Lasso

Lasso is a linear model that implements a penalty to regularize the model. Lasso is interesting because the penalty function allows for some coefficients to be set to zero, effectively removing them from the equation and thus performing a type of feature selection. On the dataset being used, Lasso will most likely perform worse than the other models simply because it is a linear model. The purpose of including it was to have some contrast to the performance of non-linear models vs linear models and show that the non-linear models are able to fit the data much better than just a linear model. ("1.1. Linear Models" (n.d.))

3.4 Model Training

The models all followed a general process for training, however some models required additional preprocessing steps that others did not. Generally, the models all were trained using the training data and were evaluated on the new testing data. The machine learning library Scikit-Learn has implementations for all the models that were considered in this project, using this implementation meant that the models could be fine-tuned and then fit to the data and once fit they could be evaluated using new data. Each model also had their own hyperparameters that would change the performance of the model when adjusted. For some models the use of GridSearch was possible because it didn't take long to fit the model or because they had fewer hyperparameters, for others, such as the SVM model, using GridSearch was just not plausible due to the time it took to produce any meaningful results, instead those models were tweaked by hand until satisfactory results were achieved.

To determine if a model improved between training iterations, the mean squared error (MSE) was observed and compared. If the MSE decreased as a result of a change made to the model, then that change would be considered good, otherwise if the MSE increased then the change that was made would be reverted. This process of examining the change in MSE would repeat until the change in MSE was not significantly increasing.

A common problem with models is that they can become overfit to the training data. This is very common in SVM and Decision Tree models just because of the nature of their algorithms. To remedy this problem the regularization technique of Bagging is used. Bagging works by taking multiple regressors, fitting them on a subset of the original data, and then aggregating the results to form a final prediction. Scikit-learn has a library that implements Bagging called BaggingRegressor that will be used with the models that require regularization.

The sections below will walk through each model's process of being trained, as well as how the models performance was determined.

3.4.1 Training SVM

The first step in training the SVM model was to standardize the given training features, this is because the SVM algorithms are not scale invariant. After the training data was standardized, the model was ready to be trained. The SVM models have a few hyperparameters that control the fit of the model, they are: the kernel, the C parameter, and the gamma. The kernel of an SVM model controls the general shape of the fit, for this model the 'rbf' kernel was used but the other kernel options were 'linear', 'poly', 'sigmoid', and 'precomputed'. The gamma parameter is the coefficient of the kernel, and the C parameter is the regularization parameter.

On the first attempt at fitting the SVM model, the resulting training error was extremely high meaning the model was completely underfit. To remedy this, the C parameter was increased to reduce the amount of regularization and this change resulted in the training error improving significantly. The gamma parameter was also adjusted slightly, however, when the gamma parameter was adjusted it caused the model to become extremely overfit to the data. To deal with the problem of overfitting, the SVM model was also used in combination with Bagging. Bagging worked on the SVM model to decrease the variance in the model and increase the bias, causing the model to perform better when tested on unseen data.

3.4.2 Training Decision Tree

The Decision Tree models have an added benefit in that they do not require any additional pre-processing of the data before it can be fit to the model. The Decision Tree models have hyperparameters: max depth, minimum samples split, and max features. The max depth is one of the more important parameters because it controls how deep the tree will grow, a tree that has no max depth will be fitted very strongly to the data, which can lead to overfitting. The minimum samples split is another parameter that is used to temper the fit of the model to the data, it controls how many samples are required to split an internal node. Finally the last hyperparemeter that was used is the max features parameter, this controls how many features to consider when looking for the best split.

The Decision Tree model was first fit without adjusting the hyperparameters, and leaving the max depth uncapped. As expected the Decision Tree was able to perfectly predict all values of the training data but performed poorly on new data, this is a result of not specifying the max depth. To adjust the overfitting of the model, the hyperparameters were adjusted, however even after adjusting the parameters it was still clear that the model was overfit. To remedy the overfitting, Bagging was used on the decision model and that substantially reduced the fit to the training data and increased the performance on new unseen data.

3.4.3 Training KNN

KNN is one of more simple models that can be used when it comes to regression. KNN is typically used for classification problems but Scikit-learn has a regression implementation of KNN that performs well. The hyperparameter used for training the KNN model was simply the number of neighbors. Increasing the number of neighbors will reduce the fit of the model to the data, while decreasing it will increase the fit. KNN can also be overfit to the data if the number of neighbors considered is small enough (i.e. close or equal to 1). The benefit of this model being so simple means that using a GridSearch algorithm to find the optimal number of neighbors is possible. Before KNN could be fit to the training data, the data had to be scaled so the model could work better.

First GridSearch was given a range of numbers from 1 to 100 which represent the number of neighbors to consider. This process returned the optimal number of neighbors for a KNN model on the data. However, the results from using KNN by itself were not that impressive, so Bagging was also used on top of KNN. Since Bagging is a regularization method and KNN has a built in regularization parameter in the form of the number of neighbors, it made logical sense to increase the flexibility of the KNN model and regularize it using Bagging. This mean't that the number of neighbors found by using GridSearch could be used as a starting point and then could be decreased to increase the fit of KNN, and at the same time tweaking the parameters of the Bagging algorithm to regularize the model. This combined approach performed better on new unseen data compared to the approach without using Bagging.

3.4.4 Training Lasso

Lasso Regression is a linear model that implements a penalty function that can set the coefficients of parameters to 0, removing them from the equation. The hyperparameter for lasso is alpha which controls

the strength of the penalty function, or in other words the strength of the regularization. Since this model is linear, it was most likely not going to perform well with the data because of how high-dimension it was. Regardless of that, it was used as a way to compare the results of a linear model to a non-linear model and see that the non-linear model is adding more value than a simple linear model. As expected the Lasso model did not perform well even with the training data, and changing the alpha hyperparameter did not effect the results either. Since the model was underfit to the training data, there was no need to use Bagging.

3.5 Final Model Building

The final stage of model training is to train the models using the best hyperparameters that were found through the optimization process of each model. The trained models will then be fit to new unseen test data. This procedure is outlined in Chapter 4.

4 Results

After the models had been optimized and trained, they could now be used to predict results for the unseen test data. The sections below will cover the results of the testing and interpretation of the results.

4.1 Performance Metrics

Since the problem of predicting future earnings after graduation is a regression problem, the class of models that were used were regression models. This also means that the metrics for assessing the performance of each model would be regression metrics. The specific metrics that were used are: R-Squared, Root Mean Square Error (RMSE), and Mean Absolute Error (MAE). The sections below will briefly explain each metric and why it was chosen.

4.1.1 R-Squared

The R-Squared metric is also known as the fraction of variance explained by the model. The best a model can achieve is an R-Squared of 1.0, this would mean that the model is able to account for 100% of the variance found in the model. The R-Squared is a very important metric because it gives us an idea of how well the model is fitting to the data. There are some limitations of the R-Squared metric and that is R-Squared will always improve or stay the same when new features are added to the model, this means that using this metric to determine the significance of features is not possible because with the addition of any feature R-Squared will not get worse, no matter the significance.

R-Squared is also an important metric but it cannot give the full picture of a regression model, so other evaluation metrics will be used alongside it.

4.1.2 RMSE

The Root Mean Square Error (RMSE) is another widely used metric. RMSE measures how far away the model's predicted values are from the real values. RMSE is very useful when trying to make accurate predictions because it can tell us how far off the model is. Another metric that was considered was the Mean Square Error (MSE), which also calculates how off the predictions were from the true values. The difference between RMSE and MSE, besides that RMSE is just the square root of MSE, is that the values reported by RMSE are in the same unit as the values being predicted, while the values reported in MSE are the unit-squared. This makes RMSE more interpretable compared to MSE which is why it was chosen. RMSE also works well in the context of the problem, since this project is about predicting the dollar earnings of a student post-graduation, it is useful to see how off the model is in terms of dollars. Since

the MSE values will be squared, it can be difficult to interpret the correlation between that error and the true values, thus RMSE was chosen.

There is a downside to using RMSE however and that is that larger outlier errors can inflate the RMSE score and thus not show as accurate of a representation of the model.

4.1.3 MAE

The Mean Absolute Error (MAE) is similar to the RMSE metric in the sense that the reported values are in the same unit as the target values. One of the main differences in the MAE is that the sign of the difference (positive or negative) does not effect the end score, instead the absolute value of the difference is taken. This means that all differences between the predicted values and the target values will linearly contribute to the resulting score. The MAE is an attractive metric because it is not as affected by outliers as MSE would be, MAE also is easy to interpret and the size of the error can be quickly related to the context of the problem at hand.

4.2 Results Table

In order to produce consistent results for the purpose of reporting, the random state = 1 was used throughout the project, this is to ensure that the results communicated here are replicable.

The table below is the program output after training and testing all the models. It should be noted that if the random state was different these results would also be different. For instance, the R-Squared of the SVM model can vary to be above or below 0.8, but generally it tends to be close to 0.8.

Each model was tested using the same unseen test data.

Table 4.1: Results from Models

Model Name	R-Squared	Root Mean Squared Error	Mean Absolute Error
SVM	0.80	4527.58	2965.03
Decision Tree	0.77	4833.70	3232.26
KNN	0.77	4911.69	3356.71
Lasso	0.09	9709.69	7705.84

4.3 Interpretation of the Results

From Table 4.1 it can be seen that SVM outperformed all models across all metrics. The worst performing model was the Lasso model which was to be expected since it is a linear model. In general, excluding the Lasso model, the models all had very similar results and performed well, but the best model was the SVM model. With regards to the data used, the models that performed best were ones that are more flexible, hence why SVM was able to perform so well. The SVM model was able to fit the data and explain 80% of

the variance, as indicated by the R-Squared value, and as a results the predictions it made had the least amount of error.

With regards to the RMSE and MAE metric for the SVM model, the MAE score is considerably lower than the RMSE. As stated previously, the MAE metric is not as affected by extreme outliers, thus the comparison of these two metrics shows that there were samples in the dataset that were outliers and may have affected the model's training. By looking at some values from the dataset, the minimum value for the target variable, earnings, is 11800.0, the maximum value is 104500.0, and the median is 31900.0. This shows that the target variable did have some outliers that may have effected how the model was fit, however the use of regularization when training the models would reduce the damage that these outliers may have caused.

Overall, the SVM model was the best performing model and provided the most accurate results. With regards to the problem of predicting earnings after graduating from a specific institution, the SVM model was able to provide an adequate and meaningful approximation of future earnings. Although the RMSE was close to $5,000, the predictions of the model are still able to convey the desired message. The purpose of the model is not to predict exactly how much a student will make, it is to be used as a tool to guide and point students in the right direction when selecting a post-secondary institution to attend. For these reasons, the results of this project are considered adequate in helping to achieve the goal of this project.

4.4 Visualization

In this section some more visualizations will be provided that help to compare the models presented above. This section will also include some interesting correlations that were discovered during the exploration of the data.

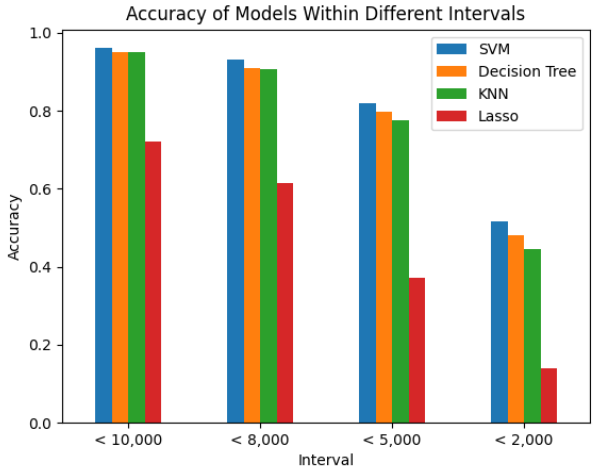


Figure 4.1: Accuracy of Models Within Different Intervals

Figure 4.1 shows compares the models and shows how accurate each model was at predicting values within the ranges, <10000, <8000, <5000, and <2000. The results in this figure do reflect the metrics of

Table 4.1 and further shows that at all intervals the SVM model was able to get the most predictions within a certain error range.

Figure 4.2 compares how each of the model's predicted values fit to the true values. The SVM, Decision Tree, and KNN models all have a similar fit to the true data, however SVM is more tightly fitted to the true values. The shape of this curve shows that the models all underestimated the true value for samples 800 to 1000. This could mean that there was some feature common in those samples that wasn't being accounted for in the training of the models.

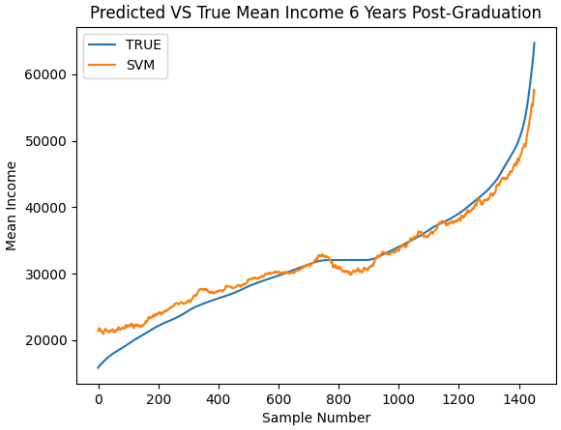
4.4.1 Interesting Correlations

This section will display some of the features that had high correlation with one another and provide an interpretation of what this correlation may mean for the greater scope of the project.

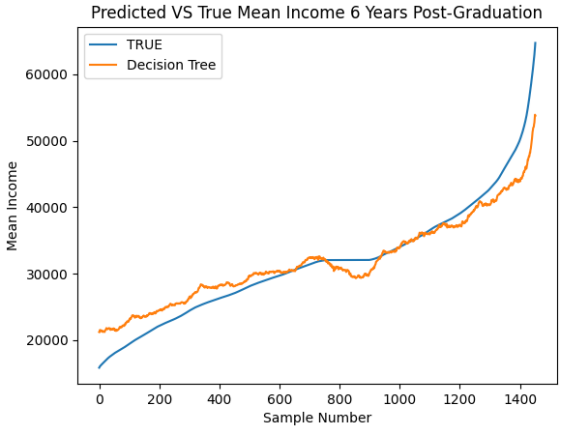
Figure 4.3a compares the completion rate of an institution against the net tuition revenue that that institution makes. This figure shows that there is a positive correlation between these two variables, and as the Net Tuition Revenue increases, so does the completion rate. This relationship could exist because, students who stay and complete their program will be paying more tuition to the institutions as opposed to the students who drop out early. It could also mean that schools that have high dropout rates will lower their tuition cost to attract more students, and thus they will suffer a loss in net tuition revenue.

Figure 4.3b compares the family income against the mean earnings 6 years after graduation. Again a positive correlation can be seen on the graph. This relationship can provide insight into one of the socio-economic issues faced today by many lower class families. Wealthy families have more resources to spend on their children and provide them with better opportunities that can lead to their children earning more after leaving school, while poorer families are unable to provide such resources and thus their children may not have the same opportunity. Institutions should take this relationship into consideration when offering equal opportunity to students who attend their schools.

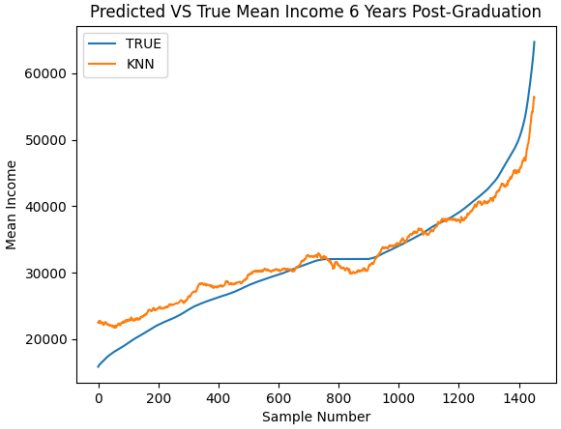
Figure 4.3c compares the withdrawal rate at an institution to the median family income. There is a negative correlation between these two variables, as the withdrawal rate increases at an institution the median family income decreases. This relationship highlights another roadblock students may face with the pursuit of post-secondary education and that is that pursuing a higher education can be very costly, and if costs get too high it may result in students being forced to withdraw. This relationship should be considered by both students and institutions, students should ensure they are financially stable enough to attend certain institutions, and institutions should try and provided resources so less fortunate students are still able to attend the school and learn without fear of being forced to withdraw.



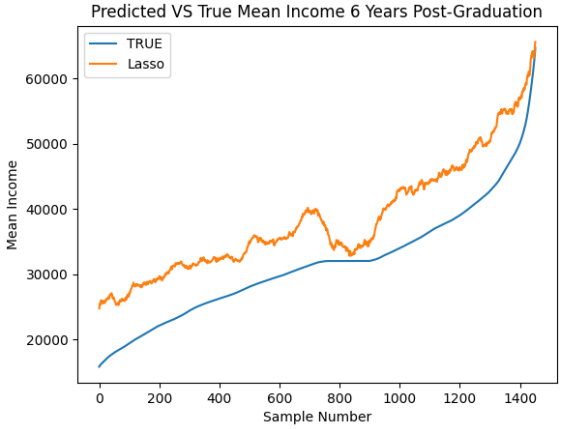
(a) SVM



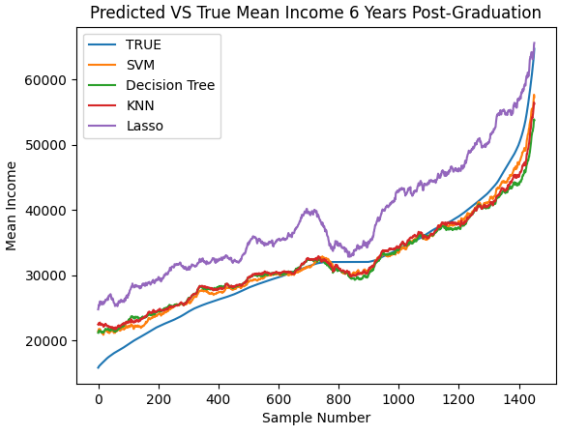
(b) Decision Tree



(c) KNN

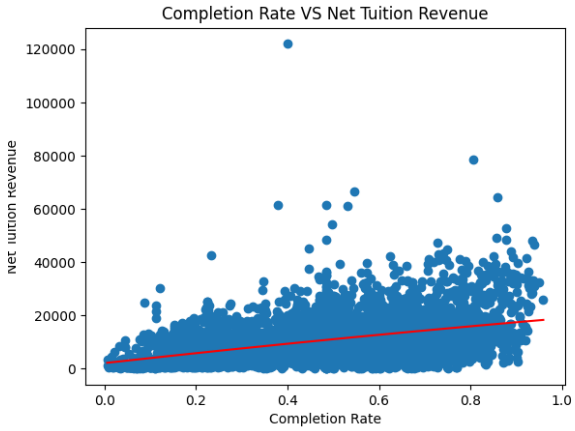


(d) Lasso

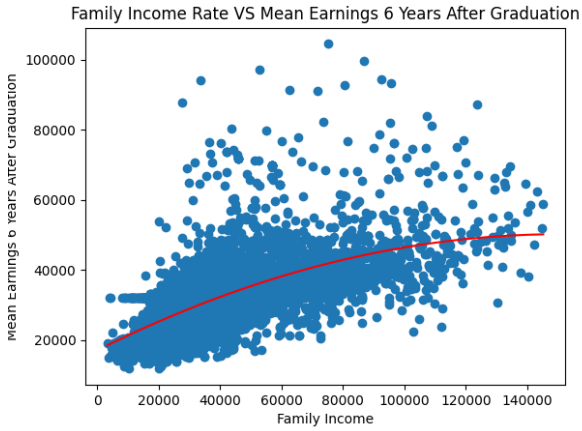


(e) All

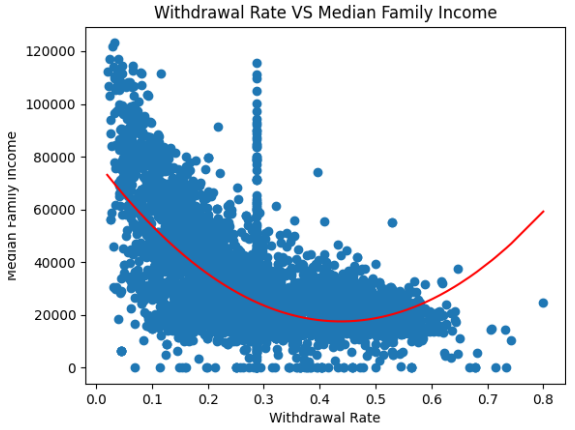
Figure 4.2: Model Fit



(a) Completion Rate VS Net Tuition Revenue



(b) Family Income VS Mean Earnings 6 Years After Graduation



(c) Withdrawal Rate VS Median Family Income

Figure 4.3: Correlations

5 Conclusion

Through the training of the models SVM, Decision Tree, KNN, and Lasso, a supervised regression model was created that was able to give accurate predictions on future earnings 6 years post-graduation. The model that proved most effective at getting results closest to the true values was the SVM model, which reported an R-squared of about 0.8 and an RMSE of 4527. Of all the predictions that the SVM model made, 80% of them were within about $5,000 of the true reported earnings, and nearly 50% were within $2,000. Correlations between features were also discovered, many of which had to do with the financial situation students were in and how that effected their outcome at these institutions.

These results show that it is possible to explain the data and provide meaningful results to the end user. Since the goal of this project is to create a tool that can guide students to picking an institution that fits their goals, the results of the model provided a good estimation that can be used by these students to guide their decisions. Although the RMSE was around $4,500 its impact on the project as a whole is not as significant. Since the data is predicting earnings that are typically in the range of tens of thousands of dollars, being off by about $4,500 is not as much of a problem because the results of the model are just for educational purposes and guiding future students decisions. It would be impossible to provide a model that perfectly predicts what a student can expect to earn after graduating, that being said it is still possible to create a model that provides a very good image of what the future might look like, and that is what this project has been able to do.

Although the results were acceptable, there are still things that need to be improved upon in future works, the first major area of improvement is the interpretability of the model. The model used around 80 features to train the data which makes it hard to understand the impact of each one, and since the model was non-linear the significance of each feature was not possible to determine. Another future area of research is better pre-processing of the data. Looking at the fit of the different models compared to the true values, there were clear patterns that all models seemed to follow. This would indicate that there may be some factor that was not being accounted for in the training process.

References

"1.1. Linear Models." n.d. *Scikit*. https://scikit-learn.org/stable/modules/linear_model.html#lasso.

"1.10. Decision Trees." n.d. *Scikit*. https://scikit-learn.org/stable/modules/tree.html#tree.

"1.4. Support Vector Machines." n.d. *Scikit*. https://scikit-learn.org/stable/modules/svm.html#svm-regression.

"1.6. Nearest Neighbors." n.d. *Scikit*. https://scikit-learn.org/stable/modules/neighbors.html#regression.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12: 2825–30.