NYC Public School Data

Layla Quinones

2022-05-18

Abstract:

There are over 1.6 million students that New York City public schools service every single year who are required to take at least one standardized test. Every year public schools receive a school quality overall rating based on a variety of achievement measures including student enrollment, and standardized test scores. In this paper, student graduation outcome data is used as features to train models to help predict a school's overall quality rating. Data was accessed from the NYC Open Data website, cleaned and transformed to create a dataset to help solve this problem, and machine learning models were used from the `caret` library in R. It was found that Support Vector Machine Classifiers and Random Forest algorithms performed the best according to accuracy metrics. Through these models, important variables that contribute to schools' overall rating were identified. It was concluded that the number of students who do and do not graduate with an Advanced Regents Diploma were good predictors of a schools overall quality rating. This study helps motivate future studies to include more student data such as demographic and economic data to help elevate the bias that this initial algorithm has due to lack of data, and to expose trends related to economic status and student outcomes.

Key words:

Education, School, Achievement, Rating, Students

Introduction:

There are approximately 1,722 schools in the New York City public school system that services over 1 million children in the community. These schools are rated based on formal school visits, feedback from students, teachers and parents from the NYC School Survey [1] and a variety of achievement measures [2]. Every year, schools receive a school quality snapshot based on the learning environment and student performance on state and city assessments. In addition, experienced educators conduct a two-day school visit where they observe classrooms and speak with parents, students, teachers, and school leaders. This is all part of a quality review report that helps identify areas of celebration and areas of focus for the school. In this paper, the overall school rating achieved via this process was used as a target variable. Data was accessed via the NYC Open Data website which houses a plethora of city data. [3]

New York City students are also required to take regents exams as part of a state learning measurement [4]. When a NYC public student takes all required regents exams they can graduate with an Advanced Regents Diploma, etc. There are variations to the high school diploma that signify various scores or programs that each student completed. These diplomas help students to get into top colleges all over the world and can signify high achievement in educational instruction. In addition data is also kept pertaining to students who take these exams and also students who enroll & graduate with specific diplomas. This data was used as predictors in the models that were trained in this paper [5].

In this paper, I sought out to use New York City student graduation outcomes data to help predict the overall school rating to help support or deny the use of student test scores as a measurement of overall school rating.

I also was interested in identifying the best type of model that could fit this data and give the most accurate predictions.

Literature review:

There are many ideas surrounding the use of test scores and student outcomes on state exams as a measure of student success to help rank schools. Some think that standardized tests are not an accurate measure of school quality, but may also be the result of community, family environment and culture in and out of school. [6] Others feel strongly that student portfolios and project based assessments are a better measurement of student success and can contribute more effectively to measuring the quality of a school. [7]

Currently there are thoughts around the correlation of school quality to the neighborhoods and community the school is in. Specifically, the idea that communities of color have a higher rate of underachieving schools than districts with more privileged demographics. [8]

Since there were no papers that were accessible to me about data analysts using models to help predict school ratings, I decided to use high school student outcomes data to see what in that data is the highest predictor of school ratings. Maybe this work would help people understand more trends that are present surrounding NYC public school ratings and student outcomes.

Currently there are no published papers about using various student data to predict overall school ratings, however there are many papers regarding standardized tests versus project based learning as a measurment of student growth.

Methodology:

The problems that were addressed in this paper are: - **1) What data categories from New York City student outcomes data contributes the most to predicting school quality ratings? - 2) Which model is the most appropriate model for using students outcome data to help predict school ratings, and what does that information tell us about overall student rating trends?**

There does not exist a single data set with variables that have student outcomes data and school ratings that is openly available to the public. In this project, data sets were found on the NYC Open Data website and merged so that they can be used in a model [3][5].

The following datasets were used to train models for this project:

- **Data Set One:** "2005 - 2020 Quality Review Ratings" - Yearly data of Quality Review ratings from 2005 to 2020 for all schools that received a Quality Review that school year, current and closed.
- **Data Set Two:** "2005 - 2010 Graduation Outcomes" - All students who first entered 9th grade in a given school year are displayed. Graduates are defined as those students earning either a Local or Regents diploma and exclude those earning either a special education (IEP) diploma or GED. Records with cohorts of 20 students or less are suppressed. August outcomes are only reported for the most recent cohort. School level results are not presented for District 79 schools, but their outcomes are included in citywide totals.

It would be interesting to find data with all regents diplomas including the special education (IEP) diploma or GED in or to minimize bias of our models. For the purposes of this paper, we will process with this data as a representation of student outcomes as defined by New York City Department of Education

Data Exploration

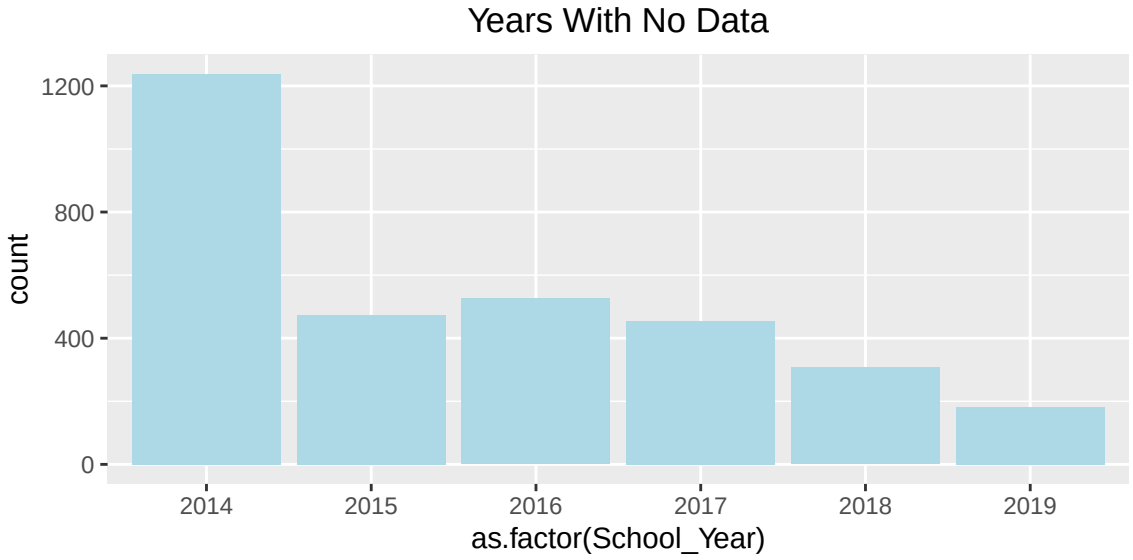
Once the data was imported into R the `select()` function from the `dplyr` package was used to select only the rows with the schools `BN - Borough/School Number` id, the `School_Year` and the `Overall_Rating` for that year. **Code Appendix A**

Data Set One

After the first inspection (below) of the data set we can see that the `School_Year` variable needed to be changed to numeric and there was one random observation that was `NA` that needed to be removed. `Overall_Rating` needed to be casted into a factor data type because it is categorical. In addition, there was a large amount of data that fell into the "No Data" category and once inspected , it was seen that yead 2014 - 2019 had no Data. This is due to the fact that these students have not graduated yet. Those years were omitted from the data set so that we only had complete cases that had ratings. **Code Appendix A**

```
##       BN             School_Year        Overall_Rating
##  Length:9009        Length:9009        Length:9009
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```
## Rows: 9,009
## Columns: 3
## $ BN             <chr> "M307", "M459", "M560", "M294", "M299", "M407", "M519",~
## $ School_Year    <chr> "2005-06", "2005-06", "2005-06", "2005-06", "2005-06", ~
## $ Overall_Rating <chr> "P", "P", "WD", "WD", "P", "WD", "WD", "P", "WD", "P", ~
```



The visualization in **Code Appendix A** shows the years in the data set that has no rating. This means that the only years we have ratings for is 2005 - 2013. In future studies I would like to get overall ratings of schools for these years to get a more current idea of the relationship between ratings and other variables such as student outcomes.

Data Set Two:

After the initial look into the dataset using `summary()` and `glimpse()` the `Cohort` column had to be cleaned to contain the year that the graduates graduated and; this was done using the `substr()` function from the `stringr` library. `stri_sub()` was also used to generate two extra columns `District` and `Borough` which is information I wanted to capture to see if it would contribute to the model. This is due to the fact that schools in New York City are located inside of a borough AND ALSO part of a district. Lastly, `Cohort` and `DBN` was changed to `School_Year` and `BN` to match with the first data set so that they can me merged together. **Code Appendix A**

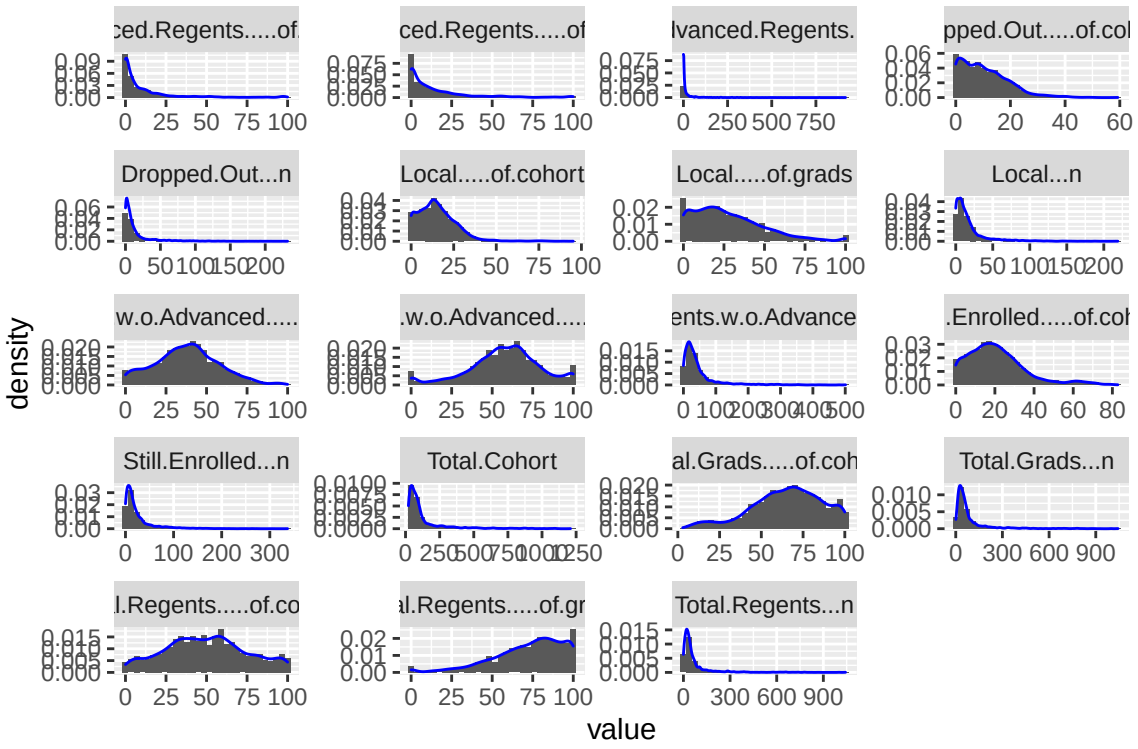
```
## Rows: 25,096
## Columns: 21
## $ DBN                                <chr> "01M292", "01M292", "01M292", "01M2~
## $ Cohort                             <chr> "2003", "2004", "2005", "2006", "20~
```

```
## $ Total.Cohort                          <int> 5, 55, 64, 78, 78, 64, 52, 87, 112,~
## $ Total.Grads...n                       <chr> "s", "37", "43", "43", "44", "46", ~
## $ Total.Grads.....of.cohort             <dbl> NA, 67.3, 67.2, 55.1, 56.4, 71.9, 6~
## $ Total.Regents...n                     <chr> "s", "17", "27", "36", "37", "32", ~
## $ Total.Regents.....of.cohort           <dbl> NA, 30.9, 42.2, 46.2, 47.4, 50.0, 3~
## $ Total.Regents.....of.grads            <dbl> NA, 45.9, 62.8, 83.7, 84.1, 69.6, 5~
## $ Advanced.Regents...n                  <chr> "s", "0", "0", "0", "0", "7", "8", ~
## $ Advanced.Regents.....of.cohort        <dbl> NA, 0.0, 0.0, 0.0, 0.0, 10.9, 15.4,~
## $ Advanced.Regents.....of.grads         <dbl> NA, 0.0, 0.0, 0.0, 0.0, 15.2, 24.2,~
## $ Regents.w.o.Advanced...n              <chr> "s", "17", "27", "36", "37", "25", ~
## $ Regents.w.o.Advanced.....of.cohort    <dbl> NA, 30.9, 42.2, 46.2, 47.4, 39.1, 2~
## $ Regents.w.o.Advanced.....of.grads     <dbl> NA, 45.9, 62.8, 83.7, 84.1, 54.3, 3~
## $ Local...n                             <chr> "s", "20", "16", "7", "7", "14", "1~
## $ Local.....of.cohort                   <dbl> NA, 36.4, 25.0, 9.0, 9.0, 21.9, 26.~
## $ Local.....of.grads                    <dbl> NA, 54.1, 37.2, 16.3, 15.9, 30.4, 4~
## $ Still.Enrolled...n                    <chr> "s", "15", "9", "16", "15", "10", "~
## $ Still.Enrolled.....of.cohort          <dbl> NA, 27.3, 14.1, 20.5, 19.2, 15.6, 3~
## $ Dropped.Out...n                       <chr> "s", "3", "9", "11", "11", "6", "1"~
## $ Dropped.Out.....of.cohort             <dbl> NA, 5.5, 14.1, 14.1, 14.1, 9.4, 1.9~
```

Training/Testing Set 1

Once both data sets were cleaned and ready `merge()` from the `dplyr` library was used for merging both data sets into one. After the initial inspection, using `summary()` and `glimpse()`, and visualizing missing variables using `gg_mis_var()` from the `naniar` package, there were some columns that needed to be changed to numeric because they were coded as characters. These are number of students and therefore should be a numeric type. **Code Appendix A**

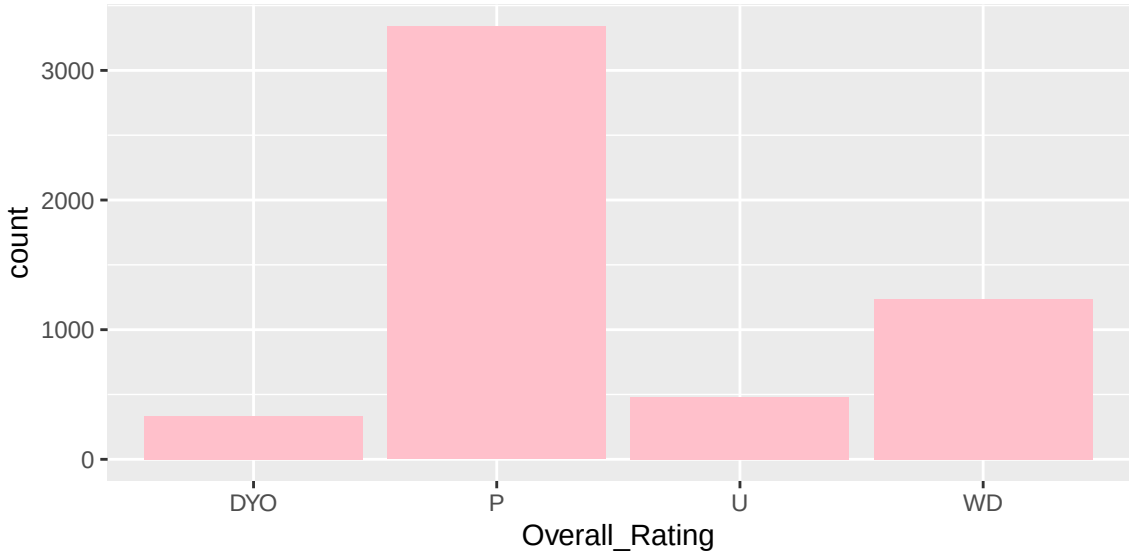
In addition, as seen from the missing data visualization above, it was discovered that there were about 2193 observations with NA values. This could be due to schools who were not included in one data set OR years that was not included in one data set. In any case, these are incomplete observations and should be removed. This may introduce bias into the model because we are omitting 30% of the rows because maybe middle school and elementary school which is not included in the data set with graduation rates. **Code Appendix A**



From the summary statistics above, we can see that some variables are highly skewed left. We will be adding a pre processing step to center and scale each variable in order to account for any issues that may arise due to skew. **Code Appendix A**

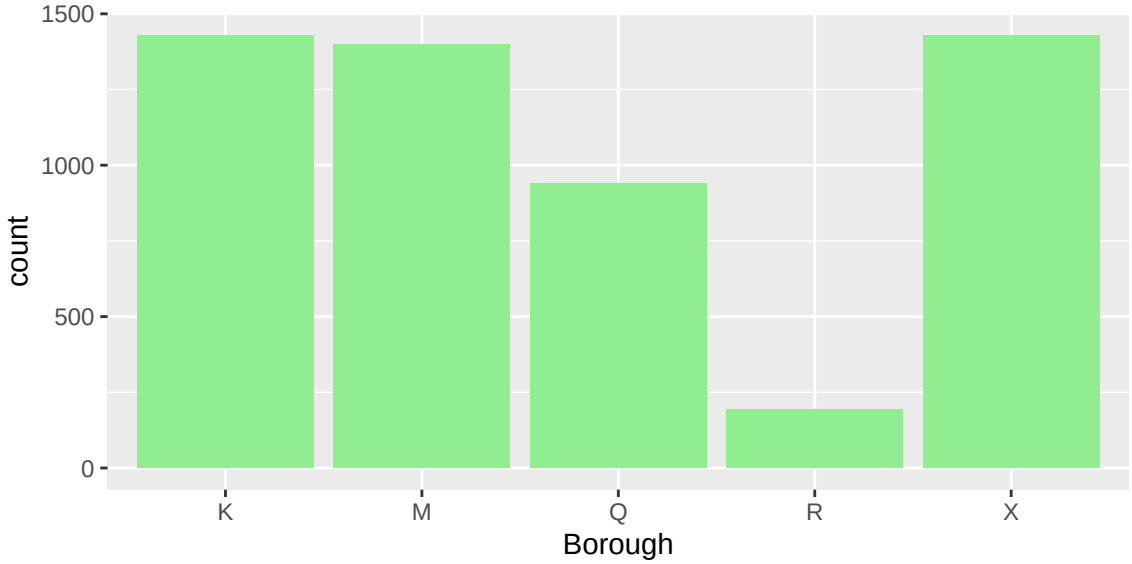
We can also see from the plots of the target variable, the distribution of data in each category. We see that there are four categories of ratings indicated by the `Overall_Rating` target variable: **Code Appendix A**

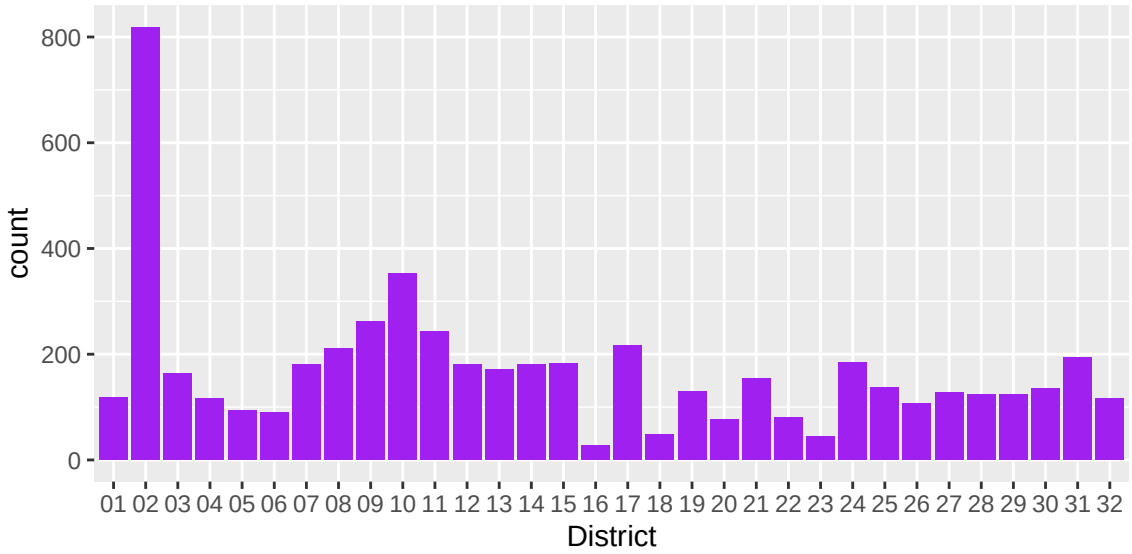
- `P` - Proficient
- `U` - Underdeveloped
- `WD` - Well Developed
- `DYO` - Data point before "No Data" began in the original data set. I chose to keep this because it exhibited a pattern and may be helpful to add to our model



From these target variable visualizations **Code Appendix A** we can get a sense of how much each category is

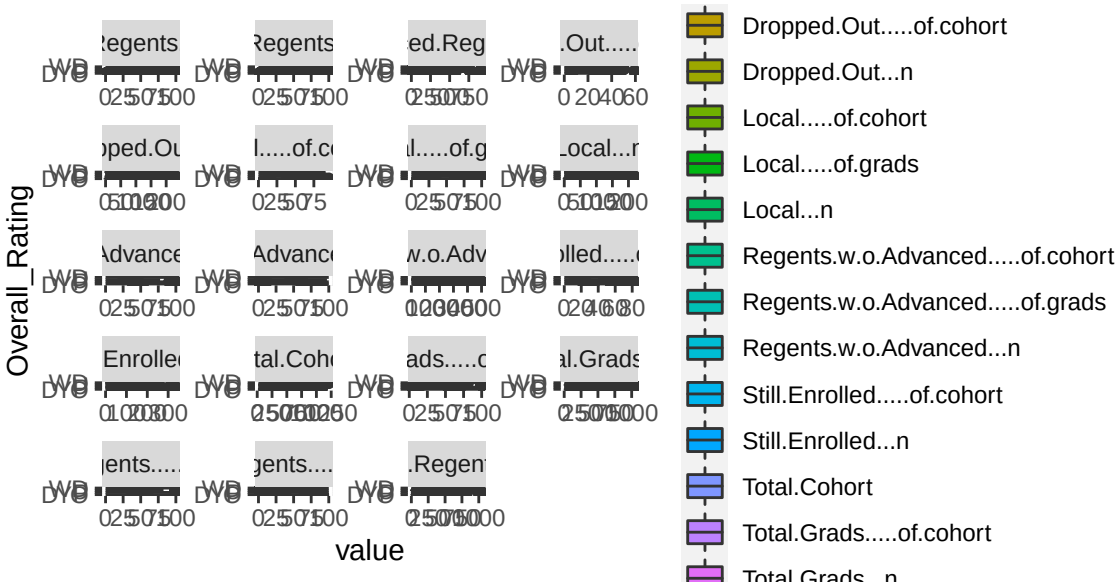
represented in each the data. We can see from the `Overall_Rating` variable above that there are a majority of school within this data set that are receiving a `P` mark where there are very few schools in the data set that received a rating of `DYO`.





From the figures in **Code Appendix A**, we can also see that there are 5 Borough Categories for Bronx(`X`), Queens(`Q`), Manhattan(`M`), Staten Island(`R`) & Brooklyn(`K`). Of course there is not a lot of representation in Staten Island, but the size of each category is related to the size of the borough and how many schools are within that borough. This is very helpful information, again, because management and administration is based on the borough, then district level. We see a similar case for the `District` where the schools in District 2 outnumber other districts due to size of district and number of schools within that district.

Next each variable was visualized with `Overall_Rating` target variable to gain some more insight into the distribution of each variable with respect to the target. As seen below, there seems to be a uniform distribution with few exceptions proportional to the number of schools that fall in each category in general. **Code Appendix A**

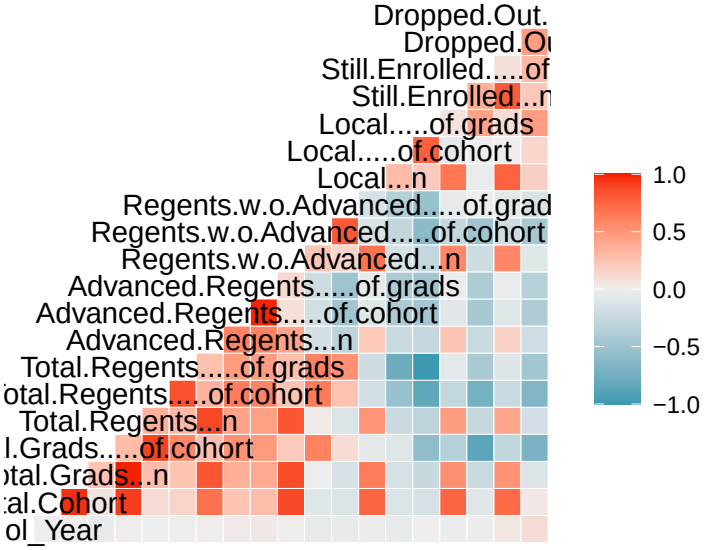


No major discrepancies were seen from the above box plots although centering and scaling the data as a pre processing step may be required for some models. What was decided as a result of this analysis was that `Overall_Rating` is actually a categorical variable in which order matters. The following changes were made to the `Overall_Rating` variable: **Code Appendix A**

- `P -> 4`
- `U - 1`
- `WD - 3`
- `DYO - 0`

Correlation: Train/Test Set 1

The next step in the EDA process included taking a look at all variables with respect to each other by using the `ggcorr` function from the `GGally` package to help visualize the correlation between variables in this data set. **Code Appendix B**

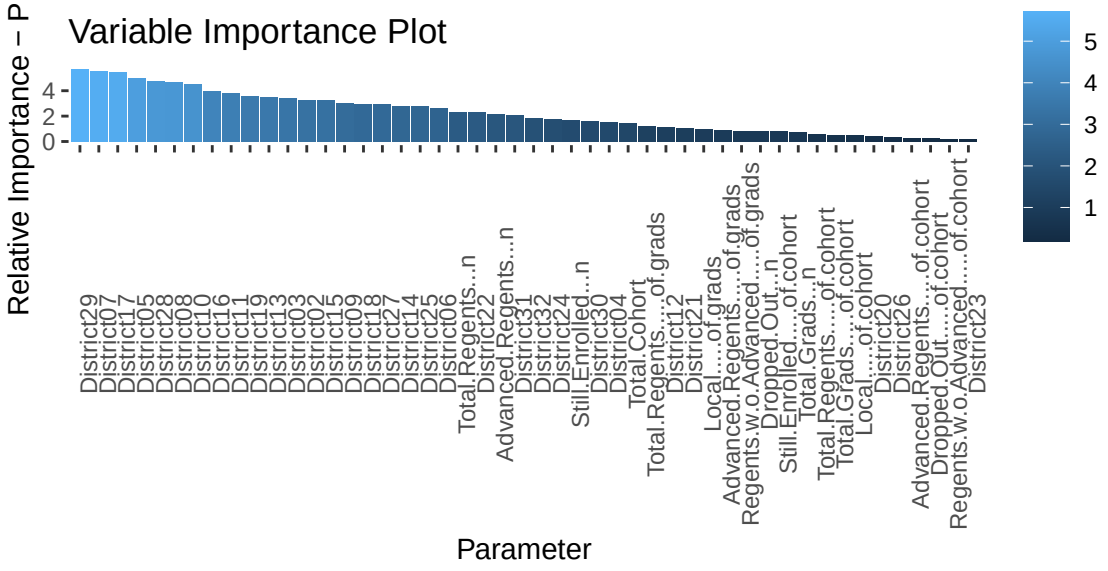


As we can see from the visualization above, there are a number of variables that are correlated to each other. This may be an issue when training certain models such as because multicollinearity makes it hard to interpret coefficients, and reduces the power of a model to identify target variables that are statistically

significant. This may mean we might want to do some feature engineering or at the very least, remove these highly correlated variables. Correlated variables and variables that are linear combinations of each other were identified using `findCorrelation()` and `findLinearCombos()` functions from the `caret` library. It was discovered that variables 2, 4, 7, 10 & 13 are linear combinations of eachother or of other variables. **Code Appendix B**

Variable Importance

Next variable importance was analyzed to help inform the decision to keep, impute or engineer some variables. As we can see from the code output below, certain variables are more important than others in our model. To achieve this, `Overall_Rating` was first set as numeric then a preliminary poisson model (`glm()`) was trained using all the variables. **Code Appendix B**



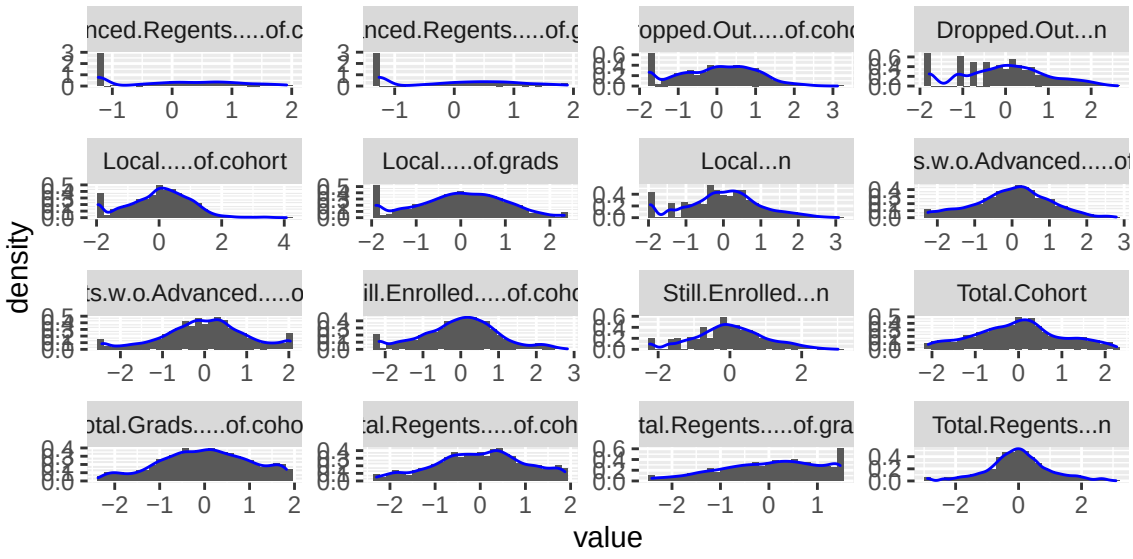
We can see from the visualization above, District is one of the highest contributing variables alongside the total number of student taking regents exams and advanced regents exams and the number of students that have dropped out of the cohort.

The multicolinearity is an issue in this case because we are looking for variable importance in the mode to help give some insight into what may contribute to certain `Overall_Ratings`. Multicolinarity will affect the `SVM` models, but will not affect the Random Forest models that will be explored. Some feature engineering steps may be explored - specifically Principal Component Analysis just to gain some information into what may yield a better model. However we'll be losing the insight that variable importance will give once we abstract the variables. Based on the variable importance and linear combinations information, variable 10, 7 and 2 will be deleted because they did not contribute much to the model and they were highly correlated linear combinations of other variables.

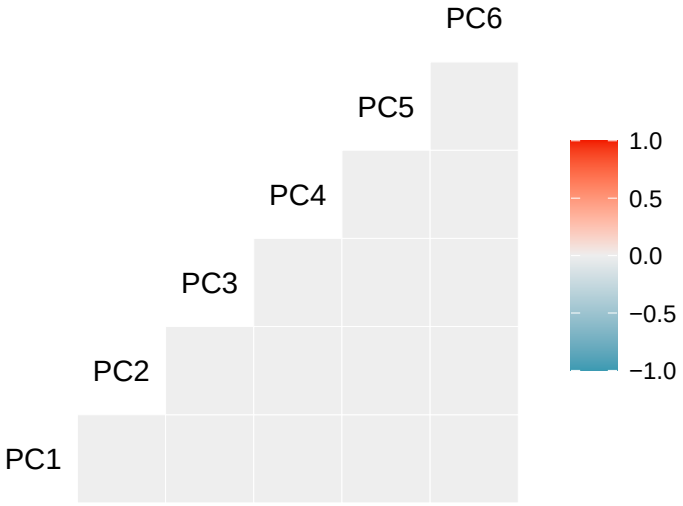
Data Preprocessing

Two data pre processing techniques were used in order to insure multicolinarity was not an issue for some models.

- `trainOne/testOne`: data set with all variables centered, normalized and Yeo Johnson transformation to the variables due to skew. Some variables were eliminated based on model results and variable importance.
- `trainTwo/TestTwo`: data set with PCA transformation - an abstraction of variables that eliminates multicolinearity



We can see the affects of the centering and scaling through the visualization above. We can see that now variables are more centered around zero and have a centered distribution that appears close to normal. **Code Appendix B**



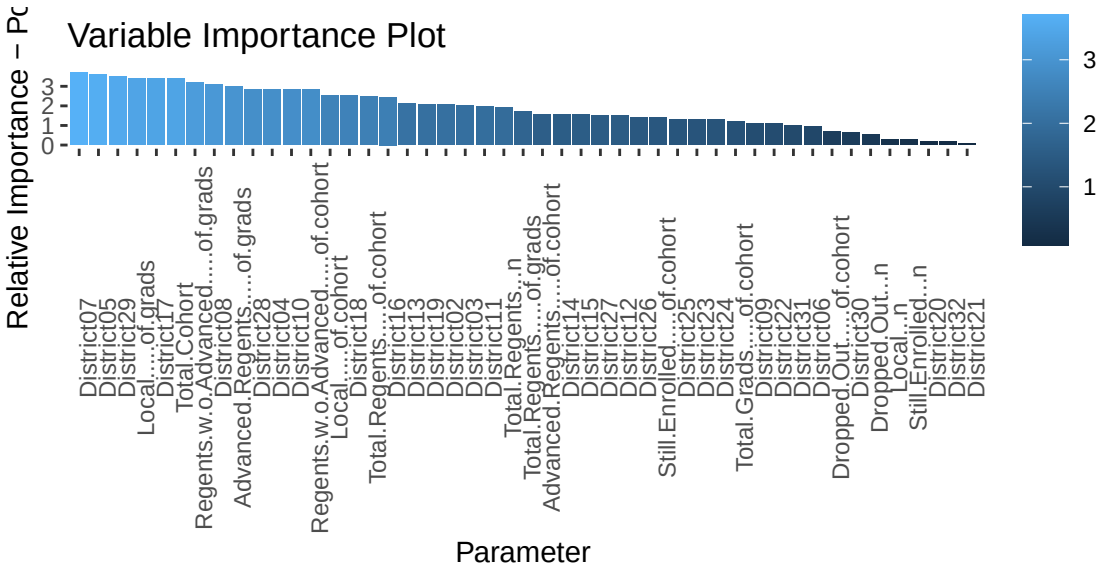
The visualization above shows the correlation heat map plot with the new variables that were created through the principal component analysis. We can see the multicolinearity has been eliminated and all varibles are lineraly independent.**Code Appendix B**

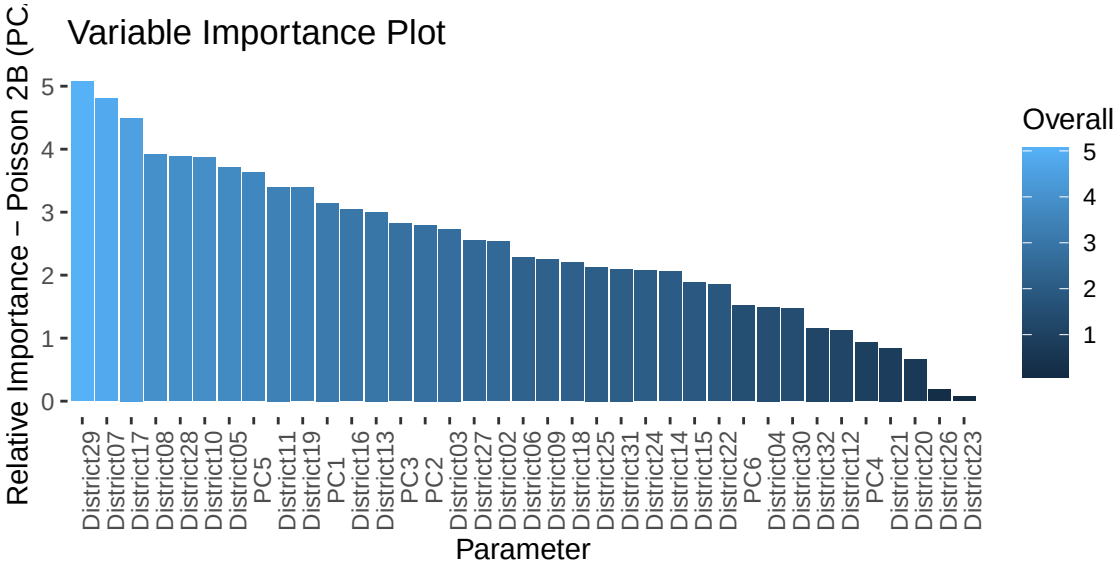
Model Building

The following models will be explored in this paper: - **Poisson**: used as a baseline and predicts count data (not meant for ranked data but used as a baseline anyway) - **SVM**: (support vector machine classifier) locates hyper planes that separates data points into categories for prediction - **Naive Bayes**: assumes independence among predictors and uses Bayes Theorem to help classify data points - **Decision Trees**: uses trees to help separate and classify data points - **Random Forest**: uses multiple decision trees to help separate and classify data points - **Neural Network**: uses a process that mimics the human brain to help recognize relationships between data points and help classify them

Poisson

The first models explored were a poisson using the `glm()` function from the `caret` library. Two models were created called `Poisson2A`(trained using non PCA variables) and `Poisson2B` (trained using PCA variables). Some relevant results are highlighted below. **Code Appendix C**





After viewing the variable importantce for each model can see that District is a consistent strong variable in both models. We can also see that maybe variables such as `PCA4` and `PCA6` and `Borough` can be eliminated because they do not contribute to this model at all. It is worth noting that the model itself was unable to use the `Borough` category and therefore other models will be explored to incorperate this category. We can also see from the table below that the models are performing similarly despite the feature engineering. We can see that the `RMSE` and `MAE` is slightly less for the first model (no PCA), however the `R^2` is much less for the second model (PCA) which may indicate there is less bias included in this model due to feature engineering. More models will be tested to see if we can correct some of the assumptions and errors made by these initial poisson models.**Code Appendix C**

Support Vector Machine Classifier (SVM)

The next set of models explored were a Support Vector Machine Classifier using the `train()` function from the `caret` library with 10 fold cross validation. Two models were created called `SVMA`(trained using non PCA variables) and `SVMB` (trained using PCA variables). Some relevant results are highlighted below. Specifically for this model, `Overall_Rating` was casted into a factor so that the algorithm classifies the category instead of estimating a range as in the poisson model. **Code Appendix D**

The above code output shows that the first model (with no PCA) achieved an accuracy of approximately 67.36 percent, whereas the model with PCA variables achieves approximately 65.56 percent. This was expected because SVM models abstract the variables anyway thereby loosing accuracy with two separate abstractions. `SVMRadial` was also applied to see if we got a better accuracy for the model.

```
##     Metric Model3_SVMLin Model4_SVMLin Model5_SVMRad Model6_SVMRad
## 1 Accuracy     0.6733034     0.6546191     0.9120901     0.8623366
```

As we can see from the accuracy measurements of each SVM explored, the third model which utilized all data variables centered, scaled and Yoe-Johnson transformed. This indicates that feature engineering is ideal in this situation. There are relationships between variables that we are not able to split apart and that SVM was able to capture. We also can see that the radial SVM was a better fit than the others which supports our hypothesis that feature engineering with hyper planes may yield better results on models. **Code Appendix D**

Naive Bayes Model

Next, a Naive Bayes Model was trained with both datasets. Using the `train(...method = "nb")` function from the `caret` library with a 10 fold cross validation so accuracy can be analyzed. Two models were created called `NBA`(trained using non PCA variables) and `NBB` (trained using PCA variables). Some relevant results are highlighted below.**Code Appendix E**

As seen in the table above, Naive Bayes model performs even worst than the baseline Poisson. This I suspect is due to the fact that Naive Bayes assumes independence of predictors. However, it was precisely for this reason that I thought NBB would perform better, because PCA illuminates the multicollinearity, but it did not. **Code Appendix E**

Decision Trees

Next, a recursive partitioning approach was taken and decision tree models were trained using `train(...method = "rpart")` function from the `caret` library with a 10 fold cross validation so accuracy can be analyzed. Two models were created called `DTA`(trained using non PCA variables) and `DTB` (trained using PCA variables). Some relevant results are highlighted below. **Code Appendix F**

```
##   Metric Model2A_Poisson Model2B_Poisson Model9_DecisionT Model10_DecisionT
## 1   RMSE    2.361615e+00    2.371470e+00       1.15663316         1.1486493
## 2    R^2    2.517702e-01    1.538501e-01       0.09010433         0.1047252
## 3    MAE    2.184758e+00    2.201750e+00       0.89167907         0.8719158
## 4    AIC    1.516444e+04    1.516444e+04               NA                NA
```

As we can see from the above table, the recursive tree method performed better than the baseline poisson method that was implemented first. When looking into these models we can see that this method identfied 2 variables as the most important: `Regents w/o Advanced` and `Total Regents` are the two most important predictors when it comes to `Overall_Rating`. Although AIC was unavailable, I would hypothesize that the AIC of this model will be better than that of the baseline poisson based on the MAE and because poisson is not typically used for count ranked data.

Random Forest

Next, a random forest model was trained using `train(...method = "rf")` function from the `caret` library with a 10 fold cross validation so accuracy can be analyzed. Two models were created called `RFA`(trained using non PCA variables) and `RFB` (trained using PCA variables). Some relevant results are highlighted below. **Code Appendix G**

```
##     Metric Model3_SVMLin Model4_SVMLin Model5_SVMRad Model6_SVMRad Model7_NB
## 1 Accuracy     0.6733034     0.6546191     0.9120901     0.8623366 0.6192363
##   Model8_NB Model9_RF Model10_RF
## 1  0.619235 0.8662009  0.7637725
```

As seen in the table above, the Random Forest models performed better than all models except for the SVM model with center, scaled and Yeo Johnson transformed variables. This is possibly due to the fact that random forest optimizes diversity of the model, and more variables can add variation and result ina better model. Still however, a Radial SVM was a better model according to these accuracy measures. In fact, the accuracy of the Radial SVM is way too good which may indicate some issues with the model.

Neural Network

The last model type that was explored was Neural Networks. The `caret` package has many neural networks in their library. Due to the high multicolinearity of the data set, neural networks with `PCA`: Principal component analysis (decomposition of variables into principal variables and maximizes variance), and `ICA`(decomposition of variables into independent underlying elements that does not focus on variance) were explored. These models were trained with the data set that does not already have the `PCA` preprocessed into it because then our results will not be meaninful if we did it twice. Three models were trained using the following functions: **Code Appendix H**

- `train(...method = "avNNet")`: baseline neural network with no feature engineering
- `train(...method = "nnet", preProcess = 'pca')`: baseline neural network with no feature engineering

After the final model was trained we can see that the SVM Radial model was the best one out of all of the (although this raises a big flag because this model we VERY accurate). Results are presented in the next section.

Discussion and Conclusions:

Twelve different models were explored in this paper to get an idea of the most important variables that contributed to the model so that we can interpret them and see what variables contribute most to overall school rating. It is also interesting to find the model that would the best fit for our data so that we can characterize this data and move forward with developing further models to predict overall school rating.

Classifiers

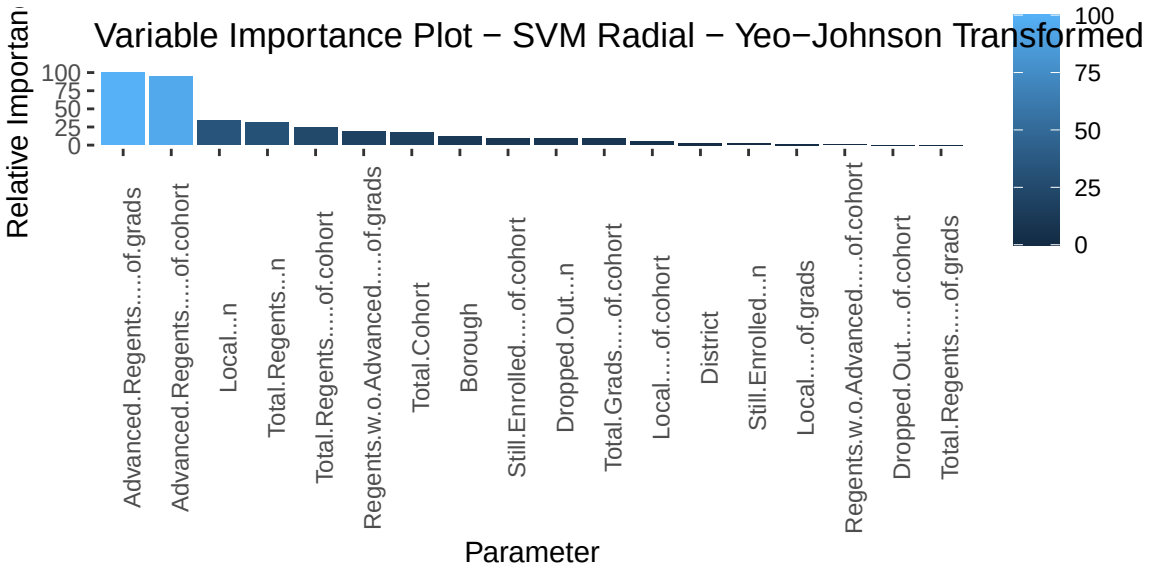
Below are the accuracy ratings for each classification model. **Code Appendx I**

```
##            Model          Accuracy
## 1         Metric          Accuracy
## 2  Model3_SVMLin 0.673303441585508
## 3  Model4_SVMLin 0.654619122013252
## 4  Model5_SVMRad 0.912090138231303
## 5  Model6_SVMRad 0.862336634237784
## 6      Model7_NB 0.619236315059591
## 7      Model8_NB 0.619234983243104
## 8      Model9_RF 0.866200917906831
## 9     Model10_RF 0.763772525927835
## 10  Model11_NNET  0.78363226431911
```

```
## 11  Model12_NNET 0.722117546199734
```

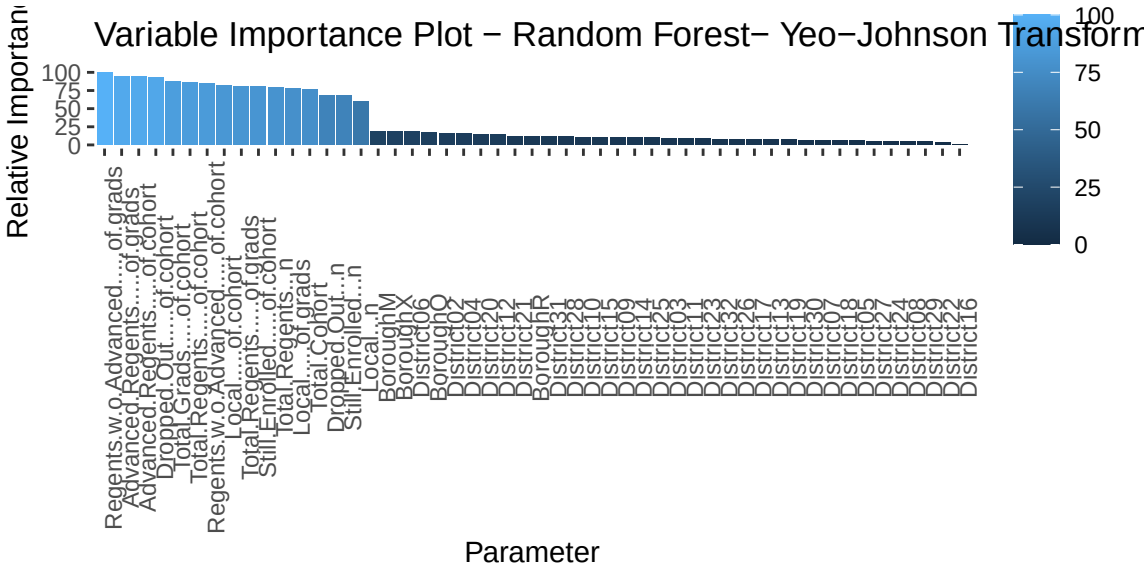
Based on the table above the best model was the Radial SVM model with the centered, scaled and Yeo Johnson transformed data which measured at approximately 91% accuracy. In my opinion this model should be rejected and SVM Radial with PCA data along with the Random Forest model on the centered and scaled data would be better models to choose in this case. Taking a deeper look into these models, lets investigate variable importance.

Below we can see a visualization of the variable importance as outlined by the SVM model. **Code Appendix J**



We can see from the graph above that the highest important variable is `Advanced.Regents..of.grads` which indicate the number of graduates who have graduated with a high school regents diploma. This means that according to the SVM model the number of students graduating with the advanced regents diploma is an important predictor for the overall rating of the school and thereby the success of that school. This can be, in part, due to the support schools receive for teachers teaching to the test and also, can be biased due to schools with higher performing students generally not needing as much support as schools with students who are not naturally doing well. There is more data that should be added to this study regarding student demographics that may give the model some helpful insight into correlations between the school location and student body to the overall rating of the school. This would serve as a better analysis to see if in fact schools are recieving higher ratings due to student outcomes rather than the community that school serves.

Below we can see a visualization of the variable importance as outlined by the Random Forest model (non PCA). **Code Appendix J**



We can see that the Random Forest Model yield different results. Here we see that the number of graduates that graduated without the advanced regents diploma is the most powerful predictor whereas, in similar fashion to the SVM presented above, the number of students who graduated with an advanced regents diploma is the second highest predictors. Regardless, both models confirm that the number of students that graduate with an advanced regent diploma is a good predictor of overall school rating. The more students who graduate with an advanced regents diploma, the higher the school rating.

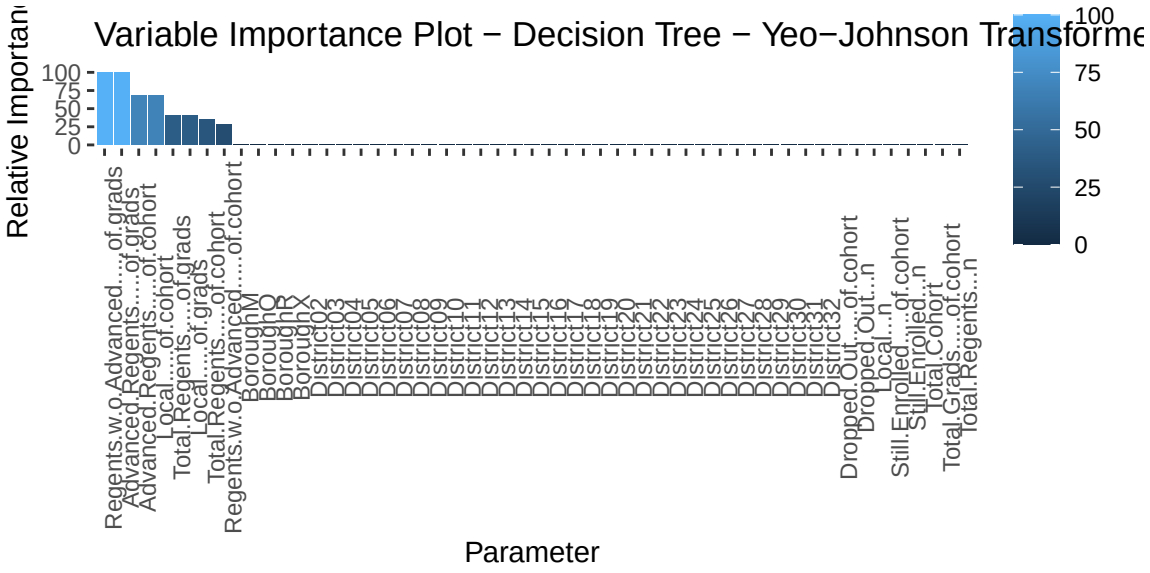
It is interesting to note above that students who dropped out is the 3rd highest predictor of school overall rating. It would be interesting to explore more about how these predictors contribute to the model. For example, can we explore the question: Does number of drop out students correlated to a poor school rating?

Regression Based

Below we can see that error metrics from all regression based models. **Code Appendix J**

```
##   Metric Model2A_Poisson Model2B_Poisson Model9_DecisionT Model10_DecisionT
## 1   RMSE    2.361615e+00    2.371470e+00       1.15663316         1.1486493
## 2    R^2    2.517702e-01    1.538501e-01       0.09010433         0.1047252
## 3    MAE    2.184758e+00    2.201750e+00       0.89167907         0.8719158
## 4    AIC    1.516444e+04    1.516444e+04               NA                NA
```

According to these error metrics, specifically the MAE value, the best regression based model is the Decision Tree model with PCA transformation. Due to the fact that the PCA transformation will not give us information into the original variables due to abstraction, below we have analyzed the variable importance of the second best model, the Decision Tree model with centered, scaled and Yeo-Johnson transformed variables. **Code Appendix J**



This model shows yet again another top variable being the students who graduated without an advanced regents diploma and a second top variable of students who graduated with an advanced regents diploma. Again all models confirm that the number of students who graduate with an advanced regents diploma is an important predictor of overall school rating.

For future work, it would be interesting to find student demographic and economic data and build models that not only incorporates students achievement data. As a subject matter expert, I would like to see if maybe those schools fall into the category of elementary school, middle school or high school as well. This is because currently there are only information about regents data for some middle schools and highschools. Elemenbtary school has been left out of this data set due to the nature of standardized test data.

Lastly, the data used in this project only ranges from 2005 - 2006 due to the merging of the two data sets. The data set with school ratings did not contain all the data for many years and therefore, we were limited. In the future, data for years 2005 - 2019 will be used to train models that pay more attention to the details of the community rather than standardized test scores.

References:

- [1] NYC School Survey. web. (n.d.). Retrieved May 16, 2022, from https://www.schools.nyc.gov/about-us/reports/school-quality/nyc-school-survey
- [2] School quality. web. (n.d.). Retrieved May 16, 2022, from https://www.schools.nyc.gov/about-us/reports/school-quality
- [3] 2005 - 2020 Quality Review Ratings. NYC Open Data. web. (n.d.). Retrieved May 16, 2022, from https://data.cityofnewyork.us/Education/2005-2020-Quality-Review-Ratings/3wfy-sn5g
- [4] N.Y.S.E.D. (2022, January 1). High School Regents Examinations. New York State Education Department. Retrieved May 16, 2022, from http://www.nysed.gov/state-assessment/high-school-regents-examinations
- [5] 2005-2010 Graduation Outcomes - School Level. NYC Open Data. web. (n.d.). Retrieved May 16, 2022, from https://data.cityofnewyork.us/Education/2005-2010-Graduation-Outcomes-School-Level/vh2h-md7a
- [6] Murrell, P. M. (2021, September 6). What Standardized Tests Do Not Measure. Rethinking Schools. Retrieved May 16, 2022, from https://rethinkingschools.org/articles/what-standardized-tests-do-not-measure/#:%7E:text=Much%20recent%20research%20on%20intelligence,of%20excellence%20or%20scholastic%20aptitude
- [7] Bloomfield, D. C. B. (2021, July 15). Opinion: Project-Based Learning Can Jumpstart a New Educational Era for NYC Schools. City Limits. Retrieved May 16, 2022, from https://citylimits.org/2

021/07/15/opinion-project-based-learning-can-jumpstart-a-new-educational-era-for-nyc-schools/
- [8] Carrie Spector, C. S. (2021, December 16). Racial disparities in school discipline are linked to the achievement. Stanford Graduate School of Education. Retrieved May 16, 2022, from https://ed.stanford.edu/news/racial-disparities-school-discipline-are-linked-achievement-gap-between-black-and-white
- [9]

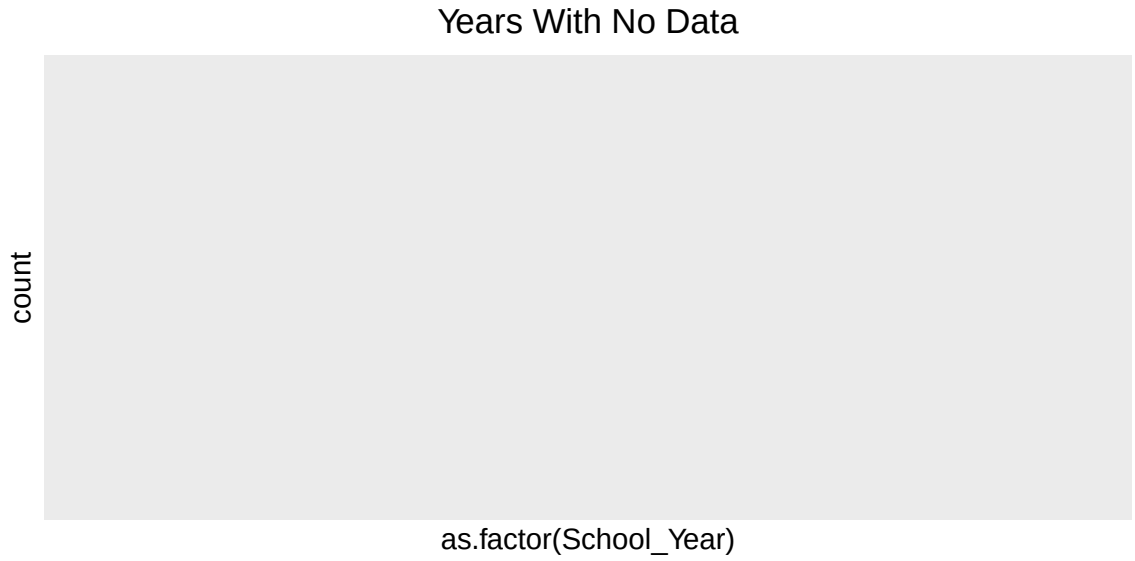
Code Appendix

Below is where the code referenced in this project can be found.

Code Appendix A

```
library(tidyr)
library(dplyr)
library(stringi)
library(caret)
library(GGally)
library(naniar)
# Load data
# Rating Data
rawOne <- read.csv("https://raw.githubusercontent.com/MsQCompSci/LQ621Final/main/2005_-_2020_Quality_Rev
#Regents Data
rawTwo <- read.csv("https://raw.githubusercontent.com/MsQCompSci/LQ621Final/main/2005-2010_Graduation_Ou

#take a glimpse at the data
summary(rawOne)
glimpse(rawOne)
```

```
#Cleaning Data Set 1
#clean the year data and make column names the same
rawOne$School_Year = as.numeric(substr(rawOne$School_Year,1,4))
#get rid of 1 NA value in School_Year
rawOne <- na.omit(rawOne, c("School_Year"))
#summary(rawOne)
rawOne$Overall_Rating <- as.character(rawOne$Overall_Rating) #target variable is a factor

#There are lots of no data values so we will take those out
noData<- rawOne[(rawOne["Overall_Rating"]== "No Data"),]
rawOne <- rawOne[!(rawOne["Overall_Rating"]== "No Data"),]
#rawOne$Overall_Rating <- droplevels(rawOne$Overall_Rating)
yearsData <- rawOne$School_Year
noyearsData <- noData$School_Year

#Years that had no data
#visualize the years with Data
ggplot(noData, aes(x = as.factor(School_Year))) +
  geom_bar(fill = "lightblue") +
  labs(title = "Years With No Data") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
#inspect rawTwo data set
summary(rawTwo)
glimpse(rawTwo)
```

```
#merge the two dataframes based on the BN and the school year
mergeOne <- merge(rawOne, rawTwo, by=c("BN","School_Year")) # 1 & 2 #data for 2005 - 2006
glimpse(mergeOne)
gg_miss_var(mergeOne)
```

```
#get rid of NA values
mergeOne<- na.omit(mergeOne, c("Total.Grads.....of.cohort"))

#there are some columns that are characters and should be numeric
mergeOne$Total.Grads...n <- as.numeric(mergeOne$Total.Grads...n)
mergeOne$Total.Regents...n <- as.numeric(mergeOne$Total.Regents...n)
mergeOne$Advanced.Regents...n <- as.numeric(mergeOne$Advanced.Regents...n)
mergeOne$Regents.w.o.Advanced...n <- as.numeric(mergeOne$Regents.w.o.Advanced...n)
mergeOne$Local...n <- as.numeric(mergeOne$Local...n)
mergeOne$Still.Enrolled...n <- as.numeric(mergeOne$Still.Enrolled...n)
mergeOne$Dropped.Out...n<- as.numeric(mergeOne$Dropped.Out...n)

# Histogram dataframe
tallDF <- mergeOne %>%
  dplyr::select(-BN, -School_Year, -Borough, -District, -Overall_Rating)%>% gather(key = 'variable', va


# Histogram plots of each variable
ggplot(tallDF) +
  geom_histogram(aes(x=value, y = ..density..), bins=30) +
  geom_density(aes(x=value), color='blue') +
  facet_wrap(. ~variable, scales='free', ncol=4)
```

```
#visualize the categorical variables including the target variables
ggplot(mergeOne, aes(x=Overall_Rating))+
  geom_bar(fill='pink')
#maybe we want to change these so that they're from greatest to least
```

```
ggplot(mergeOne, aes(x=Borough))+
  geom_bar(fill='lightgreen')

ggplot(mergeOne, aes(x=District))+
  geom_bar(fill='purple')
```

```
# Histogram dataframe
tallDF <- mergeOne %>%
  dplyr::select(-Borough, -District, -BN, -School_Year, -District, -Borough) %>%
  gather(key = 'variable', value = 'value', colnames(mergeOne)[4:22]) #omit all categorical and Bn and


# Histogram plots of each variable
ggplot(tallDF) +
  geom_boxplot(aes(x=value, y = Overall_Rating, fill = variable)) +
  facet_wrap(. ~variable, scales='free', ncol=4) #may want tofix this facet wrap
```

```
copyOne <- mergeOne
copyOne$Overall_Rating<-as.character(copyOne$Overall_Rating)
copyOne$Overall_Rating[copyOne$Overall_Rating == "P"] <- "4"
copyOne$Overall_Rating[copyOne$Overall_Rating == "U"] <- "1"
copyOne$Overall_Rating[copyOne$Overall_Rating == "WD"] <- "3"
copyOne$Overall_Rating[copyOne$Overall_Rating == "DYO"] <- "0"
#copyOne$Overall_Rating<- as.numeric(copyOne$Overall_Rating)
#copyOne$Overall_Rating<- as.factor(copyOne$Overall_Rating)
```

Code Appendex B

```
# Let's use a heat map to see the level of correlation of the numeric predictor variables.
ggcorr(mergeOne)
```

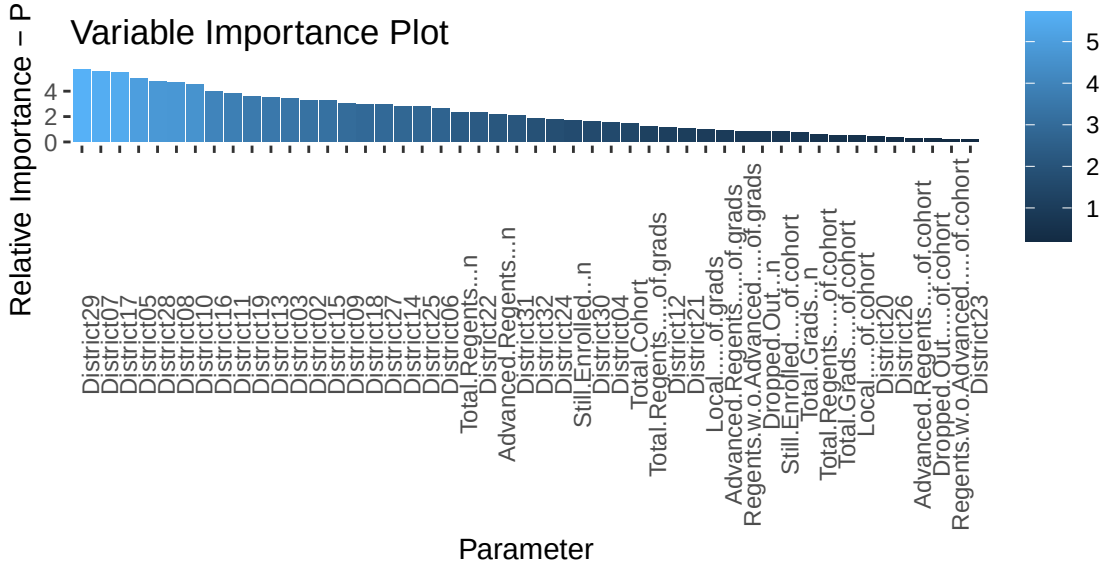
```
#FIND CORRELATED VARIABLES BY COLUMN NUMBER
contDF <- mergeOne %>%
  dplyr::select(-Borough, -District, -BN, -School_Year, -District, -Borough, -Overall_Rating)

#find correlated variables
findCorrelation(contDF, cutoff = .75, exact = FALSE)

#Find linear combinations
findLinearCombos(contDF)
```

```
colnames(contDF[10]) #DELETE
colnames(contDF[4]) # most important
colnames(contDF[7]) #one of the most important
```

```
colnames(contDF[13]) #delete
colnames(contDF[2]) #delete
colnames(contDF[4]) #most important
```

```
cleanData <- copyOne %>%
  dplyr::select(-BN,-School_Year)

#to change from character to numric or factor
cleanData$Overall_Rating<- as.numeric(copyOne$Overall_Rating)
#copyOne$Overall_Rating<- as.factor(copyOne$Overall_Rating)
```

```
# train the model
poiss1 <- glm(Overall_Rating ~ .,
              data=cleanData,
              family="poisson")

#show variable importance to help iterate on next model
#using caret and gglot
varImp(poiss1) %>% as.data.frame() %>%
    ggplot(aes(x = reorder(rownames(.), desc(Overall)), y = Overall)) +
    geom_col(aes(fill = Overall)) +
    theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.text.x = element_text(angle = 90)) +
    scale_fill_gradient() +
    labs(title = 'Variable Importance Plot',
         x = "Parameter",
         y = "Relative Importance - Poisson 1")
```




```
#normalize numerical variables
targetValue <- cleanData$Overall_Rating #save y value
#create model for normalizing
cleanDataX <- cleanData %>%
  dplyr::select(-Overall_Rating)

centerModel <-preProcess(cleanDataX, method = c("center","scale", "YeoJohnson"))

#make predictions for NA values
trainOneX <- predict(centerModel, cleanDataX)

#delete columns of linear variables identified above
trainOneX <- trainOneX %>%
  dplyr::select(-colnames(trainOneX[10]),-colnames(trainOneX[7]),-colnames(trainOneX[2]))

# Histogram dataframe
tallDF <- trainOneX %>%
```

```
  dplyr::select(-Borough, -District)%>% gather(key = 'variable', value = 'value') #omit all categorical

# Histogram plots of each variable
ggplot(tallDF) +
  geom_histogram(aes(x=value, y = ..density..), bins=30) +
  geom_density(aes(x=value), color='blue') +
  facet_wrap(. ~variable, scales='free', ncol=4)
```

```
#create model for normalizing
pcaModel <-preProcess(cleanDataX, method = c("center","scale", "pca"))

#make predictions for NA values
trainTwoX <- predict(pcaModel, cleanDataX)

#make sure there is no correlation
ggcorr(trainTwoX)
```

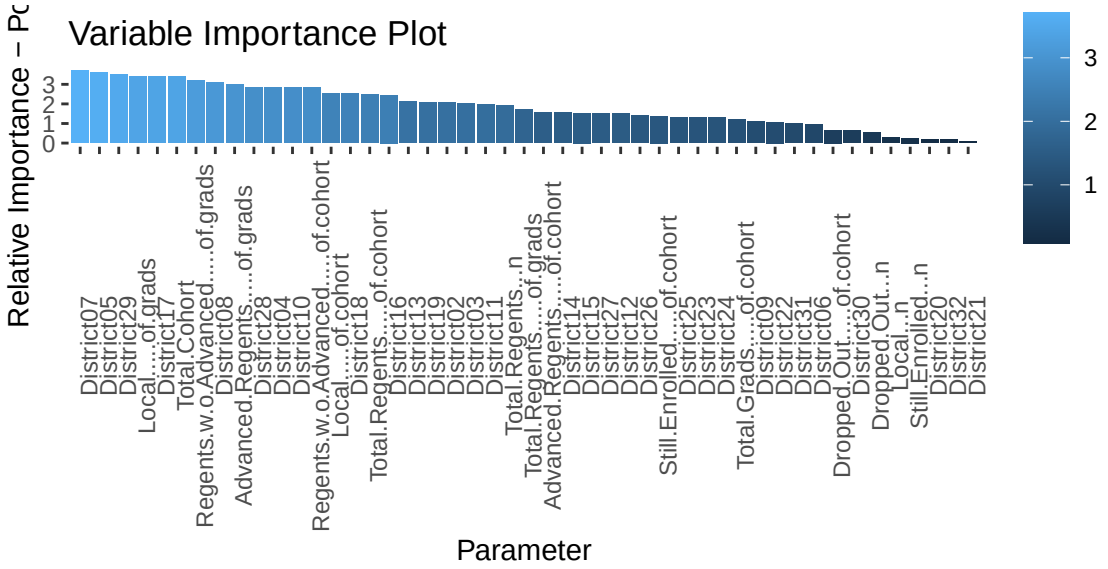
Code Appendix C

```
#add overallrating back
trainOneNew<- trainOneX
trainOneNew["Overall_Rating"] <- targetValue
trainTwoNew<- trainTwoX
trainTwoNew["Overall_Rating"] <- targetValue

# set the seed to make your partition reproducible
set.seed(369)

#get 80 percent of random indexes and separate
trainIndex <- createDataPartition(targetValue, p = .8,
                                  list = FALSE,
                                  times = 1)

#separate
trainOneNew<- trainOneNew[trainIndex, ]
testOneNew <- trainOneNew[-trainIndex, ]

#separate
trainTwoNew<- trainTwoNew[trainIndex, ]
testTwoNew <- trainTwoNew[-trainIndex, ]

#train models
#Model One (A and B)
poiss2A <- glm(Overall_Rating ~ .,data = trainOneNew, family="poisson")
#Model One
poiss2B <- glm(Overall_Rating ~ .,data = trainTwoNew, family="poisson")

summary(poiss2A)

summary(poiss2B)

#Training Metircs (not needed)
#separate variables and target
```

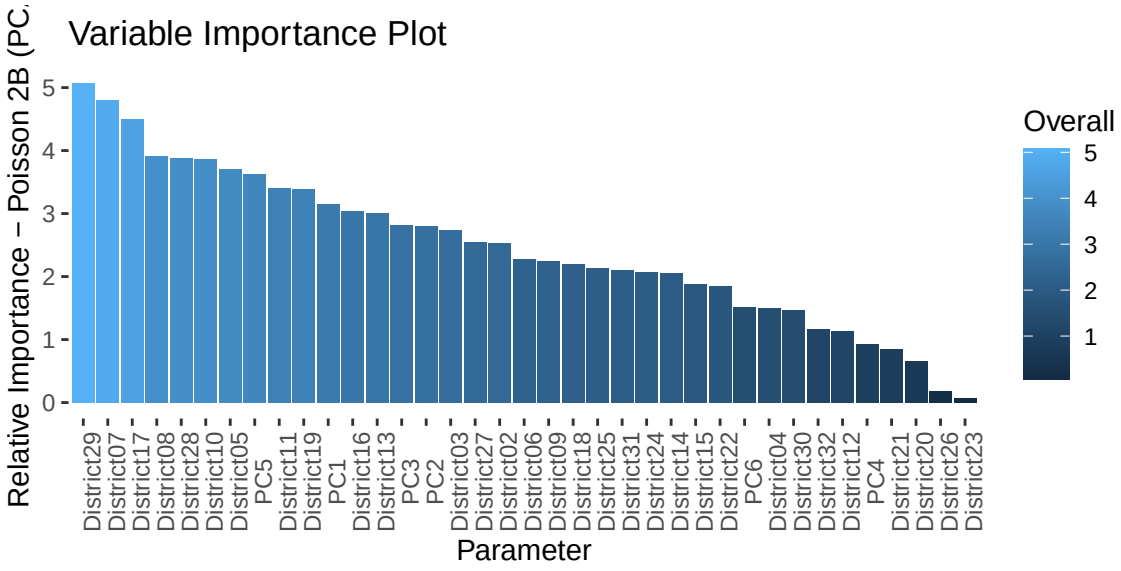
```
#trainOneX <- trainOneNew %>% dplyr::select(-Overall_Rating)
#trainOneY <- trainOneNew %>% dplyr::select(Overall_Rating)

#separate variables and target
#trainTwoX <- trainTwoNew %>% dplyr::select(-Overall_Rating)
#trainTwoY <- trainTwoNew %>% dplyr::select(Overall_Rating)

# Evaluate Model 1 with train data
#predYAtrain <- predict(poiss2A, newdata=trainOneX)
#modelResultsTrainOne <- data.frame(obs = trainOneY, pred=predYAtrain)
#colnames(modelResultsTrainOne) = c('obs', 'pred')

# This grabs RMSE, Rsquaredand MAE by default
#modelEvalTrain2A <- defaultSummary(modelResultsTrainOne)

#evaluate model 1 using test data
testOneX <- testOneNew %>% dplyr::select(-Overall_Rating)
testOneY <- testOneNew %>% dplyr::select(Overall_Rating)

# Evaluate Model 1 with train data
predYtest2A <- predict(poiss2A, newdata=testOneX)
modelResultsTest2A <- data.frame(obs = testOneY, pred=predYtest2A)
colnames(modelResultsTest2A) = c('obs', 'pred')

# This grabs RMSE, Rsquaredand MAE by default
modelEvalTest2A <- defaultSummary(modelResultsTest2A)

#display RMSE, Rsuared and MAE
#modelEvalTest2A


#evaluate model 2 using test data
testTwoX <- testTwoNew %>% dplyr::select(-Overall_Rating)
testTwoY <- testTwoNew %>% dplyr::select(Overall_Rating)

# Evaluate Model 1 with train data
predYtest2B <- predict(poiss2B, newdata=testTwoX)
modelResultsTest2B <- data.frame(obs = testTwoY, pred=predYtest2B)
colnames(modelResultsTest2B) = c('obs', 'pred')

# This grabs RMSE, Rsquaredand MAE by default
modelEvalTest2B <- defaultSummary(modelResultsTest2B)

#display RMSE, Rsuared and MAE
#modelEvalTest2B
```

```
#show variable importance to help iterate on next model
#using caret and gglot
varImp(poiss2A) %>% as.data.frame() %>%
    ggplot(aes(x = reorder(rownames(.), desc(Overall)), y = Overall)) +
    geom_col(aes(fill = Overall)) +
    theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.text.x = element_text(angle = 90)) +
```

```
    scale_fill_gradient() +
    labs(title = 'Variable Importance Plot',
         x = "Parameter",
         y = "Relative Importance - Poisson 2A")
```




```
#show variable importance to help iterate on next model
#using caret and gglot
varImp(poiss2B) %>% as.data.frame() %>%
    ggplot(aes(x = reorder(rownames(.), desc(Overall)), y = Overall)) +
    geom_col(aes(fill = Overall)) +
    theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.text.x = element_text(angle = 90)) +
    scale_fill_gradient() +
    labs(title = 'Variable Importance Plot',
         x = "Parameter",
         y = "Relative Importance - Poisson 2B (PCA)")
```



```
ErrorsDF <- data.frame(matrix(ncol = 5, nrow = 4))
ErrorsDF["Metric"] <- c("RMSE", "R^2", "MAE", "AIC")
ErrorsDF["Model2A_Poisson"]<- rbind(as.data.frame(modelEvalTest2A),"AIC" = as.data.frame(AIC(poiss2A))[
ErrorsDF["Model2B_Poisson"]<- rbind(as.data.frame(modelEvalTest2B),"AIC" = as.data.frame(AIC(poiss2A))[
ErrorsDF %>% dplyr::select(Metric, Model2A_Poisson, Model2B_Poisson)
```

Code Appendix D

```
#train models
#set up tunning grid
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

#Model One (A and B)
SVMA <- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method = "svmLinear",
                trControl=trctrl,
                preProcess = c("center", "scale"),
                tuneLength = 10)
#Model One
SVMB <- train(as.factor(Overall_Rating) ~ .,data = trainTwoNew, method = "svmLinear",
                trControl=trctrl,
                preProcess = c("center", "scale"),
                tuneLength = 10)

# Evaluate Model 1 with trEST data
predYtestSVMA <- predict(SVMA,  newdata=testOneX)
modelResultsTestSVMA  <- data.frame(obs = testOneY, pred=predYtestSVMA)

# Evaluate Model 1 with train data
predYtestSVMB <- predict(SVMB, newdata=testTwoX)
modelResultsTestSVMB <- data.frame(obs = testTwoY, pred=predYtestSVMB)
```

```
SVMA
SVMB
```

```
#Model One (A and B)
SVMC <- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method = "svmRadial",
                trControl=trctrl,
                preProcess = c("center", "scale"),
                tuneLength = 10)
#Model One
SVMD <- train(as.factor(Overall_Rating) ~ .,data = trainTwoNew, method = "svmRadial",
                trControl=trctrl,
                preProcess = c("center", "scale"),
                tuneLength = 10)

# Evaluate Model 1 with train data
predYtestSVMC <- predict(SVMC,  newdata=testOneX)
modelResultsTestSVMC  <- data.frame(obs = testOneY, pred=predYtestSVMC)
colnames(modelResultsTestSVMC) = c('obs', 'pred')

# Evaluate Model 1 with train data
predYtestSVMD <- predict(SVMD, newdata=testTwoX)
modelResultsTestSVMD <- data.frame(obs = testTwoY, pred=predYtestSVMD)
colnames(modelResultsTestSVMD) = c('obs', 'pred')
```

```
AccDF <- data.frame(matrix(ncol = 5, nrow = 1))
AccDF["Metric"] <- c("Accuracy")
AccDF["Model3_SVMLin"]<- SVMA$results$Accuracy
AccDF["Model4_SVMLin"]<- SVMB$results$Accuracy
AccDF["Model5_SVMRad"]<- SVMC$results$Accuracy[10]
AccDF["Model6_SVMRad"]<- SVMD$results$Accuracy[10]
AccDF %>% dplyr::select(-X1, -X2, -X3,-X4,-X5)
```

Code Appendix E

```
#train models
#Model Three (A and B)
NBA <- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method="nb", trControl = trainControl(me
#Model One
NBB<- train(as.factor(Overall_Rating) ~ .,data = trainTwoNew, method="nb", trControl = trainControl(met

AccDF <- data.frame(matrix(ncol = 5, nrow = 1))
AccDF["Metric"] <- c("Accuracy")
AccDF["Model3_SVMLin"]<- SVMA$results$Accuracy
AccDF["Model4_SVMLin"]<- SVMB$results$Accuracy
AccDF["Model5_SVMRad"]<- SVMC$results$Accuracy[10]
AccDF["Model6_SVMRad"]<- SVMD$results$Accuracy[10]
AccDF["Model7_NB"]<- NBA$results$Accuracy[2]
AccDF["Model8_NB"]<- NBB$results$Accuracy[2]
AccDF %>% dplyr::select(-X1, -X2, -X3,-X4,-X5)
```

```
NBA
NBB
```

Code Appendix F

```
#train models
#Model Three (A and B)
DTA <- train(Overall_Rating ~ .,data = trainOneNew, method="rpart", trControl = trainControl(method = '
#Model One
DTB<- train(Overall_Rating ~ .,data = trainTwoNew, method="rpart", trControl = trainControl(method = 'c
#display models
DTA
DTB
```

```
ErrorsDF <- data.frame(matrix(ncol = 5, nrow = 4))
ErrorsDF["Metric"] <- c("RMSE", "R^2", "MAE", "AIC")
ErrorsDF["Model2A_Poisson"]<- rbind(as.data.frame(modelEvalTest2A),"AIC" = as.data.frame(AIC(poiss2A))[
ErrorsDF["Model2B_Poisson"]<- rbind(as.data.frame(modelEvalTest2B),"AIC" = as.data.frame(AIC(poiss2A))[
ErrorsDF["Model9_DecisionT"]<- c(DTA$results$RMSE[1],DTA$results$Rsquared[1], DTA$results$MAE[1], NA)
ErrorsDF["Model10_DecisionT"]<- c(DTB$results$RMSE[1],DTB$results$Rsquared[1], DTB$results$MAE[1], NA)
ErrorsDF %>% dplyr::select(Metric, Model2A_Poisson, Model2B_Poisson, Model9_DecisionT, Model10_Decision
```

Code Appendix G

```
#train models
#Model Five (A and B)
```

```
mtry <- sqrt(ncol(trainOneNew))
tunegrid <- expand.grid(.mtry=mtry)

RFA <- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method="rf", metric = "Accuracy", tuneGri
#Model One
RFB <- train(as.factor(Overall_Rating) ~ .,data = trainTwoNew, method="rf", metric = "Accuracy", tuneGri

#display models
RFA
RFB
```

```
#Accuracy Table
AccDF <- data.frame(matrix(ncol = 5, nrow = 1))
AccDF["Metric"] <- c("Accuracy")
AccDF["Model3_SVMLin"]<- SVMA$results$Accuracy
AccDF["Model4_SVMLin"]<- SVMB$results$Accuracy
AccDF["Model5_SVMRad"]<- SVMC$results$Accuracy[10]
AccDF["Model6_SVMRad"]<- SVMD$results$Accuracy[10]
AccDF["Model7_NB"]<- NBA$results$Accuracy[2]
AccDF["Model8_NB"]<- NBB$results$Accuracy[2]
AccDF["Model9_RF"]<-RFA$results$Accuracy
AccDF["Model10_RF"]<-RFB$results$Accuracy
AccDF %>% dplyr::select(-X1, -X2, -X3,-X4,-X5)
```

Code Appendix H

```
#neural network models
#Baseline fits a bunch of nnet models and finds the best one
NnetA<- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method="avNNet", metric = "Accuracy",

#Model with PCA
NnetB <- train(as.factor(Overall_Rating) ~ .,data = trainOneNew, method="nnet", metric = "Accuracy",

#dpsiplay model
NnetA
NnetB
```

```
#Accuracy table
AccDF <- data.frame(matrix(ncol = 5, nrow = 1))
AccDF["Metric"] <- c("Accuracy")
AccDF["Model3_SVMLin"]<- SVMA$results$Accuracy
AccDF["Model4_SVMLin"]<- SVMB$results$Accuracy
AccDF["Model5_SVMRad"]<- SVMC$results$Accuracy[10]
AccDF["Model6_SVMRad"]<- SVMD$results$Accuracy[10]
AccDF["Model7_NB"]<- NBA$results$Accuracy[2]
AccDF["Model8_NB"]<- NBB$results$Accuracy[2]
AccDF["Model9_RF"]<-RFA$results$Accuracy
AccDF["Model10_RF"]<-RFB$results$Accuracy
AccDF["Model11_NNET"]<-NnetA$results$Accuracy[9]
AccDF["Model12_NNET"]<-NnetB$results$Accuracy[9]
AccDF %>%
  dplyr::select(-X1, -X2, -X3,-X4,-X5)%>%
  gather(key = 'Model', value = 'Accuracy')
```

Code Appendix I

```
AccDF <- data.frame(matrix(ncol = 5, nrow = 1))
AccDF["Metric"] <- c("Accuracy")
AccDF["Model3_SVMLin"]<- SVMA$results$Accuracy
AccDF["Model4_SVMLin"]<- SVMB$results$Accuracy
AccDF["Model5_SVMRad"]<- SVMC$results$Accuracy[10]
AccDF["Model6_SVMRad"]<- SVMD$results$Accuracy[10]
AccDF["Model7_NB"]<- NBA$results$Accuracy[2]
AccDF["Model8_NB"]<- NBB$results$Accuracy[2]
AccDF["Model9_RF"]<-RFA$results$Accuracy
AccDF["Model10_RF"]<-RFB$results$Accuracy
AccDF["Model11_NNET"]<-NnetA$results$Accuracy[9]
AccDF["Model12_NNET"]<-NnetB$results$Accuracy[9]
AccDF %>%
  dplyr::select(-X1, -X2, -X3,-X4,-X5)%>%
  gather(key = 'Model', value = 'Accuracy')
```

Code Appendix J

```
#using caret and gglot
imp<- varImp(SVMC)[1]

#using caret and gglot
imp$importance %>%
    ggplot(aes(x = reorder(rownames(.), desc(Overall)), y = Overall)) +
    geom_col(aes(fill = Overall)) +
    theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.text.x = element_text(angle = 90)) +
    scale_fill_gradient() +
    labs(title = 'Variable Importance Plot - SVM Radial - Yeo-Johnson Transformed',
         x = "Parameter",
         y = "Relative Importance")
```

```
#using caret and gglot
imp<- varImp(RFA)[1]

#using caret and gglot
imp$importance %>%
    ggplot(aes(x = reorder(rownames(.), desc(Overall)), y = Overall)) +
    geom_col(aes(fill = Overall)) +
    theme(panel.background = element_blank(),
          panel.grid = element_blank(),
          axis.text.x = element_text(angle = 90)) +
    scale_fill_gradient() +
    labs(title = 'Variable Importance Plot - SVM Radial - Yeo-Johnson Transformed',
         x = "Parameter",
         y = "Relative Importance")
```

```
ErrorsDF <- data.frame(matrix(ncol = 5, nrow = 4))
ErrorsDF["Metric"] <- c("RMSE", "R^2", "MAE", "AIC")
ErrorsDF["Model2A_Poisson"]<- rbind(as.data.frame(modelEvalTest2A),"AIC" = as.data.frame(AIC(poiss2A))[
ErrorsDF["Model2B_Poisson"]<- rbind(as.data.frame(modelEvalTest2B),"AIC" = as.data.frame(AIC(poiss2A))[
```

```
ErrorsDF["Model9_DecisionT"]<- c(DTA$results$RMSE[1],DTA$results$Rsquared[1], DTA$results$MAE[1], NA)
ErrorsDF["Model10_DecisionT"]<- c(DTB$results$RMSE[1],DTB$results$Rsquared[1], DTB$results$MAE[1], NA)
ErrorsDF %>%
  dplyr::select(Metric, Model2A_Poisson, Model2B_Poisson, Model9_DecisionT, Model10_DecisionT)
```