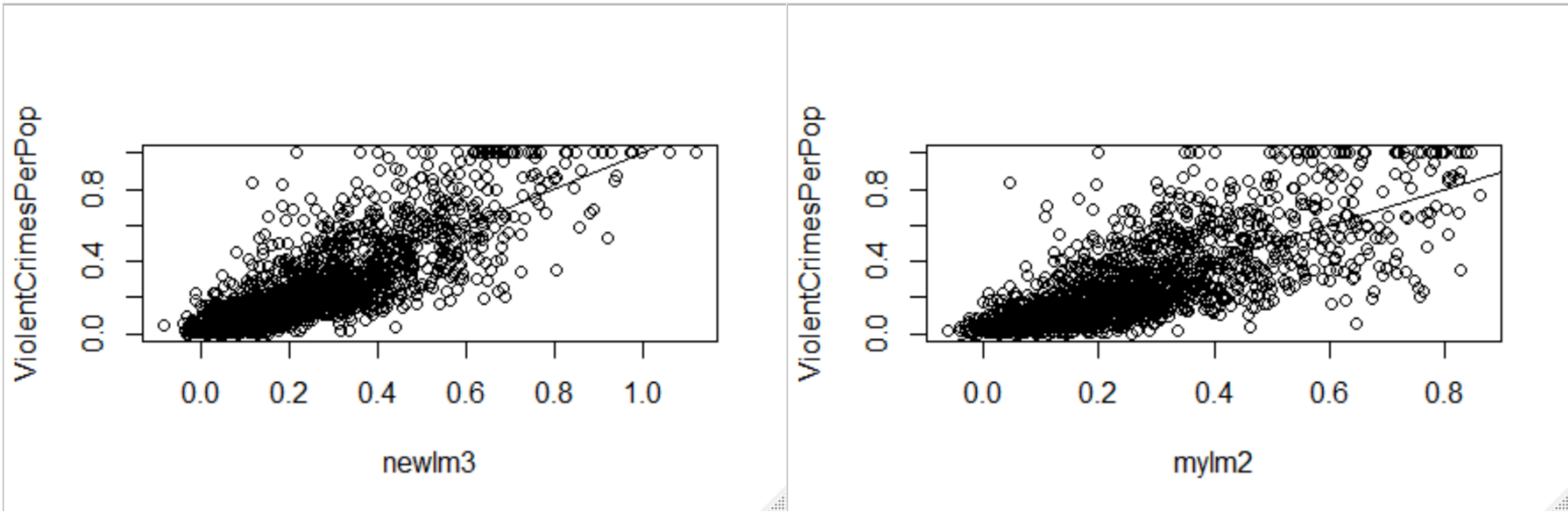
Group 24
12/13/2019

Project Final Draft

We worked on the crime data set, which consists of 128 socio-economic and police variables. The first five variables are identifiers, so they were not used in our data analysis: state, county, community, communityname, and fold. There are 1994 rows where each row corresponds to a different community, and 128 columns in the data set where each column represents a type of demographic data. Our goal with the crime data set is to use various data analysis methods to predict the number of violent crimes per one-hundred-thousand people. The question we are trying to answer is: based on variables such as a community's living conditions and police activity, can we predict a definite number of violent crimes per a population of one-hundred-thousand? This question is extremely important because once we know which factors produce a certain amount of crime per 100,000 people, we can find remedies to reduce the rate of crime in that community. Methods we used to find these factors include plotting variables against violentcrimeperpop, using a for loop to find correlation between variables and violentcrimeperpop, and comparing the mean squared errors of training and test sets, and other explorative methods. We attempted to find graphs that showed a clear indication of both x and y-variables depending on each other, and on achieving a lower mean squared error value. Finding the variables that have less spread (lower variance) when plotted with violent crime, and being able to predict the number of violent crime from those variables found, allow us as a society to not only change the communities that have high crime rates, but implement the same changes everywhere else despite some areas not needing them. This would ensure crime would drop dramatically, and crime would be very rare in places that had low crime rates initially.

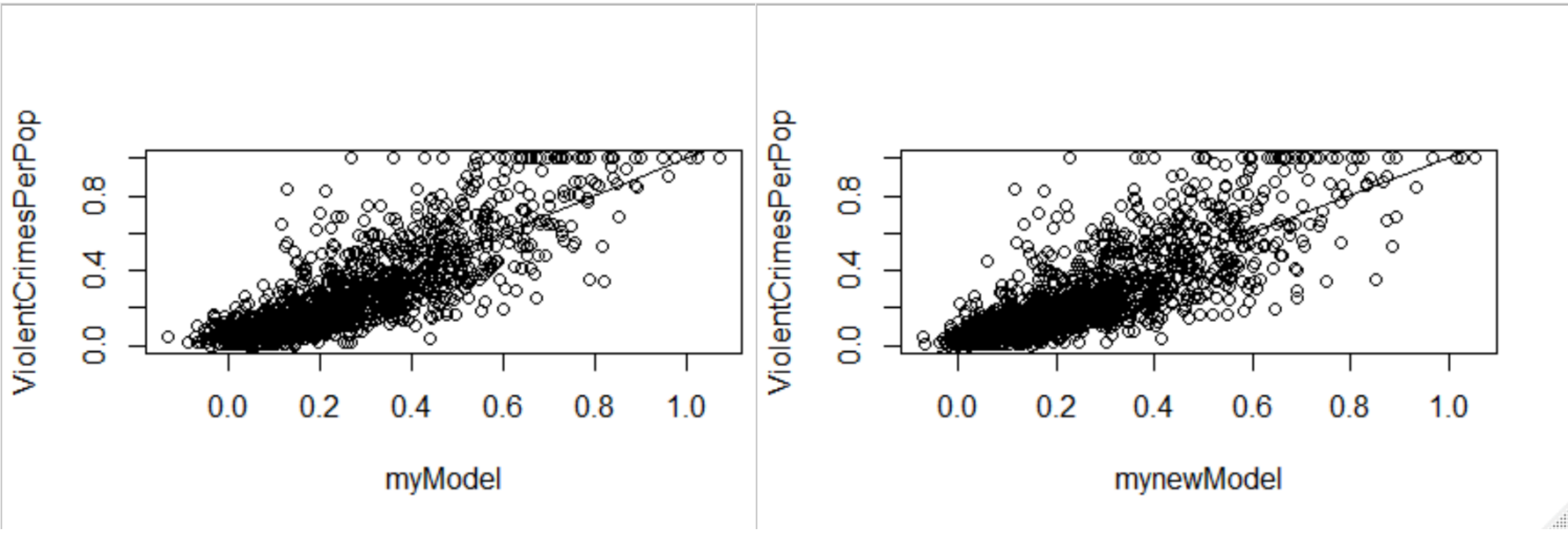
Since the goal of the crime data set is to predict an amount of crime per one-hundred-thousand people using data analysis methods, that means the crime data set works with regression, and not classification. Solving regression is easier than solving classification because we don't have to find definite answers as to why something occurs or doesn't occur (1 or 0). We just find factors that bring about something like crime, and based on our models, decide whether the value we're trying to predict is correct or not. We needed to find ways of presenting data so that one can easily look at our code or linear models and see what variables are to blame for more violent crime. The first step in doing so was to plot each variable against the column we're trying to predict. However, most of the columns had ommitted data, which we fixed by creating a for loop and replacing each "N/A" with the mean of that column. Doing so changes all the non-numeric data in the data set into numbers that do not affect the column's mean. Now, we can start using the function 'pairs()' with a few variables at a time plotting against violentcrimeperpop. However, this method of attempting to see correlation very quickly is very tedious. Therefore, our next step was to create a for loop to sift from variable 6 to variable 127 using the cor function. We knew that cor has a range of -1 to 1, with the latter meaning there is a definite correlation between the x and y variable. So for the if statement's condition, we set it so the for loop prints out the index of a column if it has a high correlation value with violentcrimeperpop. We made sure to choose cor value as 0.5 so that it would output us less than 20 indexes, so that we don't overfit our linear models. Once we had our eight variables found from our cor for loop, we then used 10-fold cross validation to see the overall mse of the linear model using these eight variables. We would use this overall mse to compare with the mean squared errors of the training and test sets later on. Next, we created a linear model with ViolentCrimePerPop as our y-variable, and columns 6-127 as our x-variables all added together. Using the 'summary()' function along with the 'plot()' and "cor()" functions, we would decide whether to discard variables based on their P-values and or whether they had stars. The diagram on the left below (newlm3), is of the simplified linear model that started with variables 6-127. We also created a simplified linear model for the variables found by the cor for loop, which we plotted against violentcrimeperpop (mylm2).



newlm3 is the simplified* lm of all variables 6-127 added together as the x-value, that includes only the best 3-star variables.
mylm2 is the simplified* lm of mylm1.

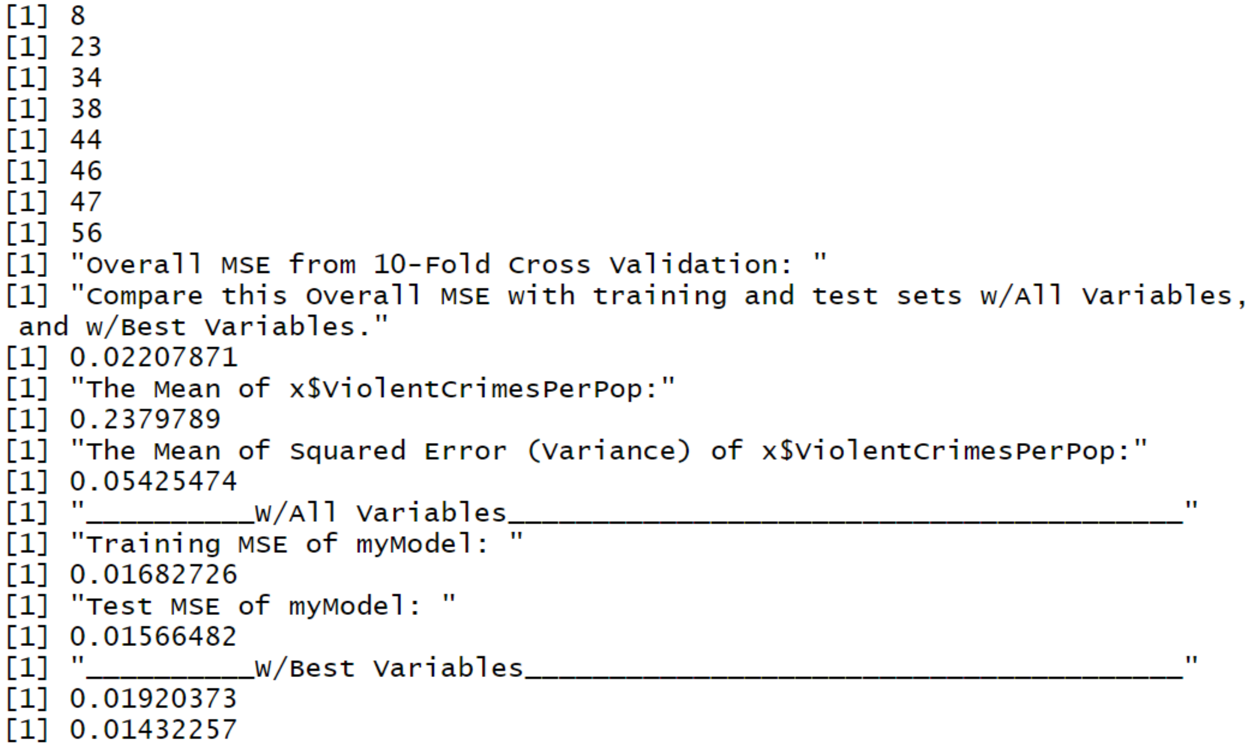
*Simplified means took out variables that have high P-value, no stars, or showed no correlation when plotting.

One can argue that the graph on the left has less variance and mean squared error than the graph on the right. However, the graph on the right has more dots centered around the 'abline(a=0, b=1)' line than the left graph. We can conclude that the model on the right is good, but we need to find further ways to reduce that spread and error. Next, we then started using 'pairs()' and plotting each individual variable with violentcrimeperpop to verify whether the variables we have left are worth keeping. We then start creating training and test sets. We outputted the training mean squared error of the linear model that had variables 6-127 as our x-variables (myModel), then outputted the test mse. And we did the same for the linear model with the best three star variables mixed in with the best three star variables found from the cor for loop (mynewModel). In the R console, we can then compare the initial overall mse with the training and test mse of the two linear models we have left.



myModel is the lm that has all variables 6-127 added together as the x-value. mynewModel is the simplified* lm of myModel w/simplified cor variables added together as the x-value.

*Simplified means took out variables that have high P-value, no stars, or showed no correlation when plotting.



```
[1] 8
[1] 23
[1] 34
[1] 38
[1] 44
[1] 46
[1] 47
[1] 56
[1] "Overall MSE from 10-Fold Cross Validation: "
[1] "Compare this Overall MSE with training and test sets w/All Variables,
 and w/Best Variables."
[1] 0.02207871
[1] "The Mean of x$ViolentCrimesPerPop:"
[1] 0.2379789
[1] "The Mean of Squared Error (Variance) of x$ViolentCrimesPerPop:"
[1] 0.05425474
[1] "__________W/All Variables________________________________________"
[1] "Training MSE of myModel: "
[1] 0.01682726
[1] "Test MSE of myModel: "
[1] 0.01566482
[1] "__________W/Best Variables_______________________________________"
[1] 0.01920373
[1] 0.01432257
```

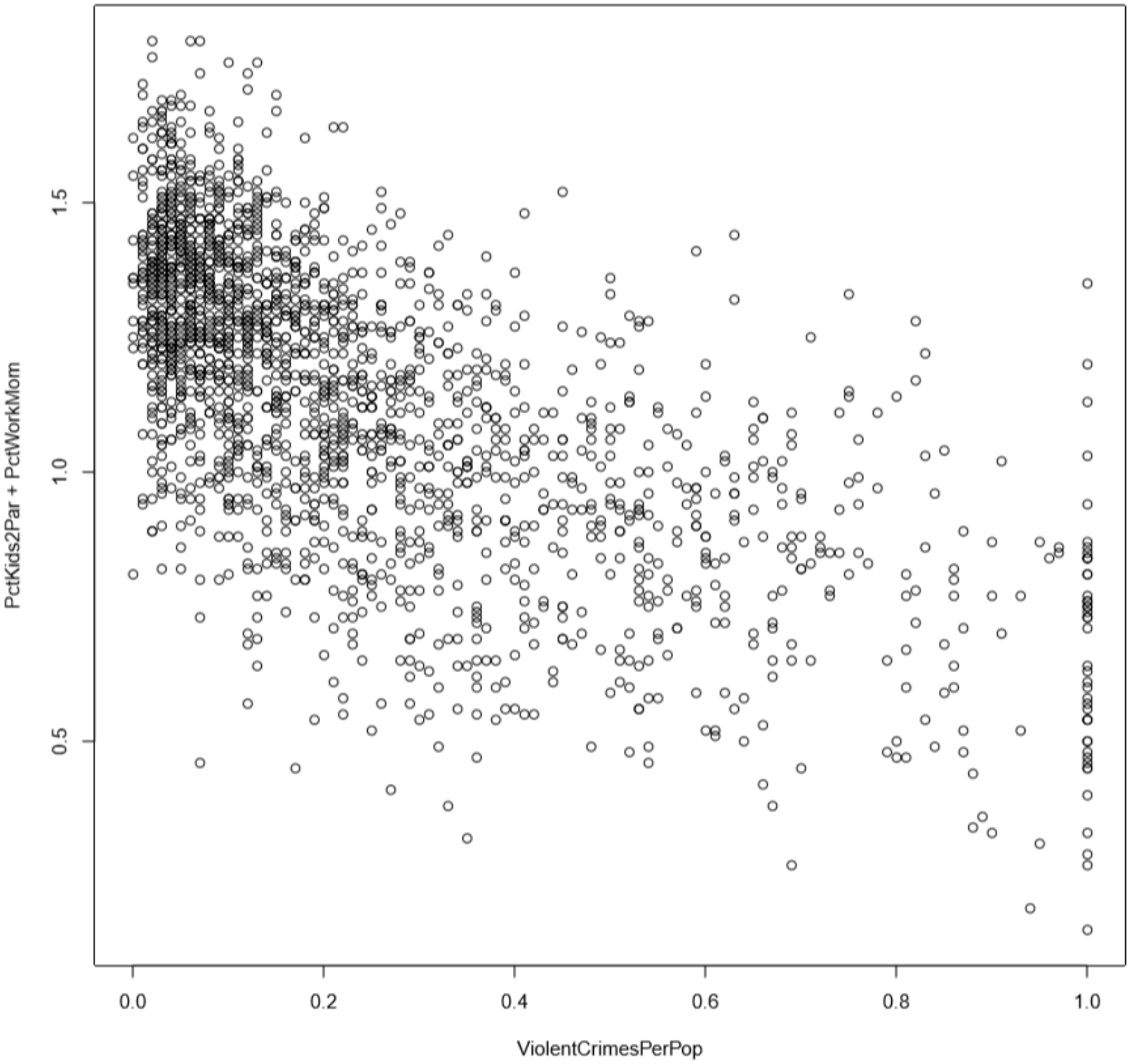
R Console log of Output

The plot on the right (myModel) is a good model because we've achieved even less spread, thus a lower mean squared error than what we had initially. You can verify this by looking at the console log above, at the overall mse in the beginning, versus the test set mse at the very bottom. We successfully lowered the mean squared error by about 0.007 which shows a significant change. What we have now is an even more accurate model that could be used to predict violent crime per a population of one-hundred-thousand. As for other methods we tried, we tried creating three separate plots where we added only similar looking plots to each other. The reason we did this was so we don't have one graph that's a big moshpit of factors that lead to violent crime. This way, we know three different groups of factors that can lead to violentcrimeperpop. If you run the R code we submitted, it'll export files Top.pdf, Middle.pdf, and Bottom.pdf. You can also see these diagrams at the very bottom of this report.

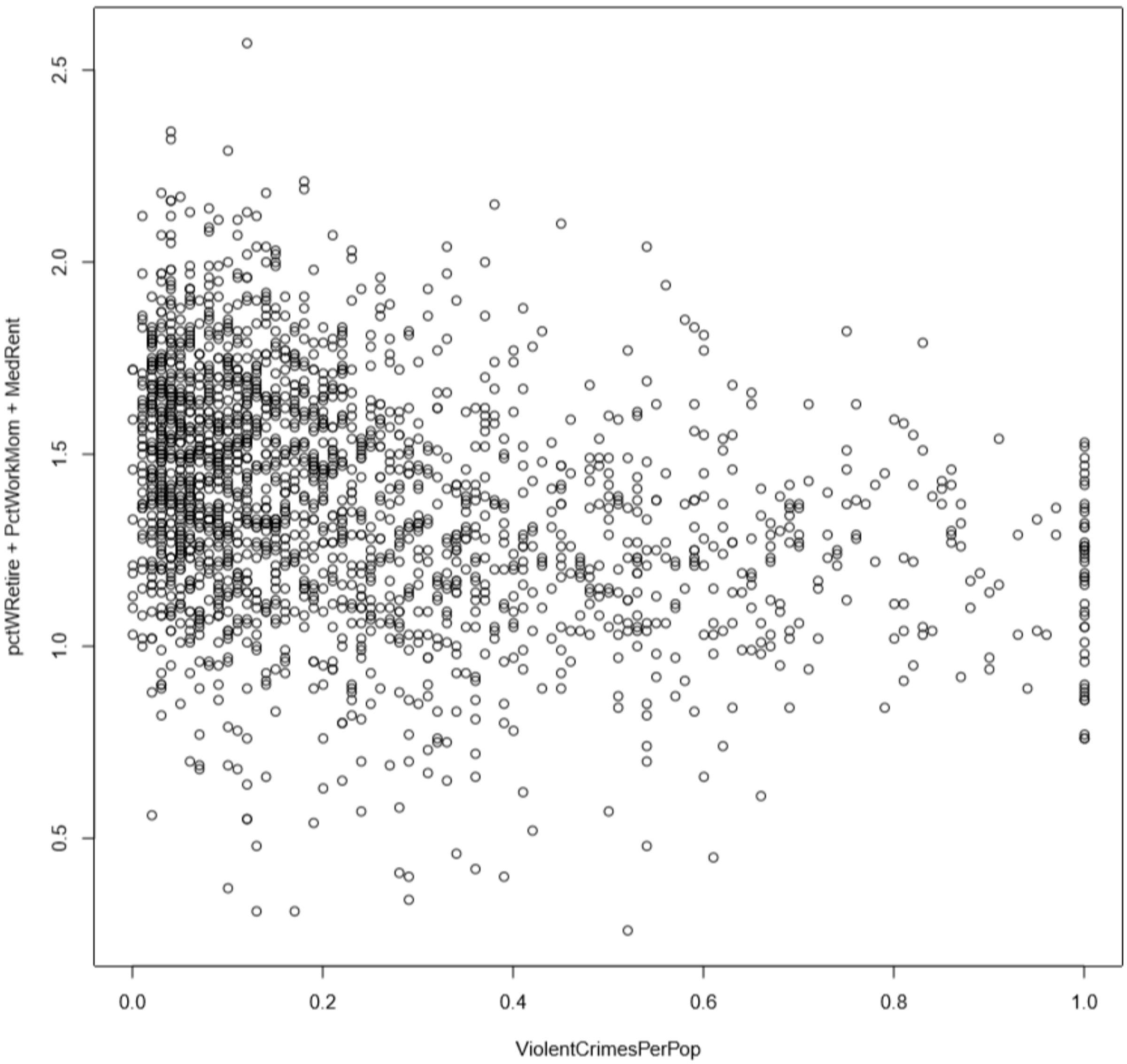
Our models show a correlation between the variables and the column we are trying to predict. The exported PDFs: Top, Middle, and Bottom show this, as well as the plots inside R if you click left and right in the Plots tab module. The only plot we have that has a big variance is pctUrban when plotted with ViolentCrimesPerPop as the x-value. We believe pctUrban is a variable that shouldn't be thrown away and not used, because urban areas are where there are a lot of people, and so it makes sense for there to be more crime in towns or cities. So even though pctUrban is the only questionable variable, we still think we have the correct variables, as shown through our models in R, in the paragraphs above, and exported PDFs.

While doing this project, we found that the pairs function was not helpful in the beginning. There are 128 variables, so even if we plotted 122 together with pairs, we would be unable to read what was produced. Producing a dozen PDFs of the pairs of ten variables at a time also was not helpful, because many variables wouldn't be compared to other variables. And even though we implemented the training and test set, we're unsure what they mean for us. For a test set MSE, sometimes it would jump higher than 1 and we have no idea why it does that (fourth line above the console line). We've been comparing the MSEs of the training and test set with the MSE of ViolentCrimesPerPop, but the training and test set MSE almost always is lower than the MSE of ViolentCrimesPerPop. So does this mean through our training and test set, we were able to achieve even better results and accuracy, by sampling randomly in data set x? How would not knowing what we sampled help us here? Something that worked out well was our loops. Our for loops, like the one used to find correlation in the beginning, was crucial to us finding variables that work well with ViolentCrimesPerPop. In the end, combining those variables with the three star variables found from the simplified linear model reduced the spread. Another loop helped replace all the omitted data with the mean of their column. That allowed us to use R functions like 'pairs()' and 'mean()', because now we have data that is all numeric. The linear models and the usage of R function 'summary()' with the linear models allowed us to find our three star variables, which in the end allowed us to produce models where our variables correlated with what we're trying to find. For the plots, we've noticed on the x and y-axis, the numbers go from 0 to 1, but we need to predict the number of violent crimes per 100,000 people, so we don't know how to find that number if the axises are from 0 to 1.

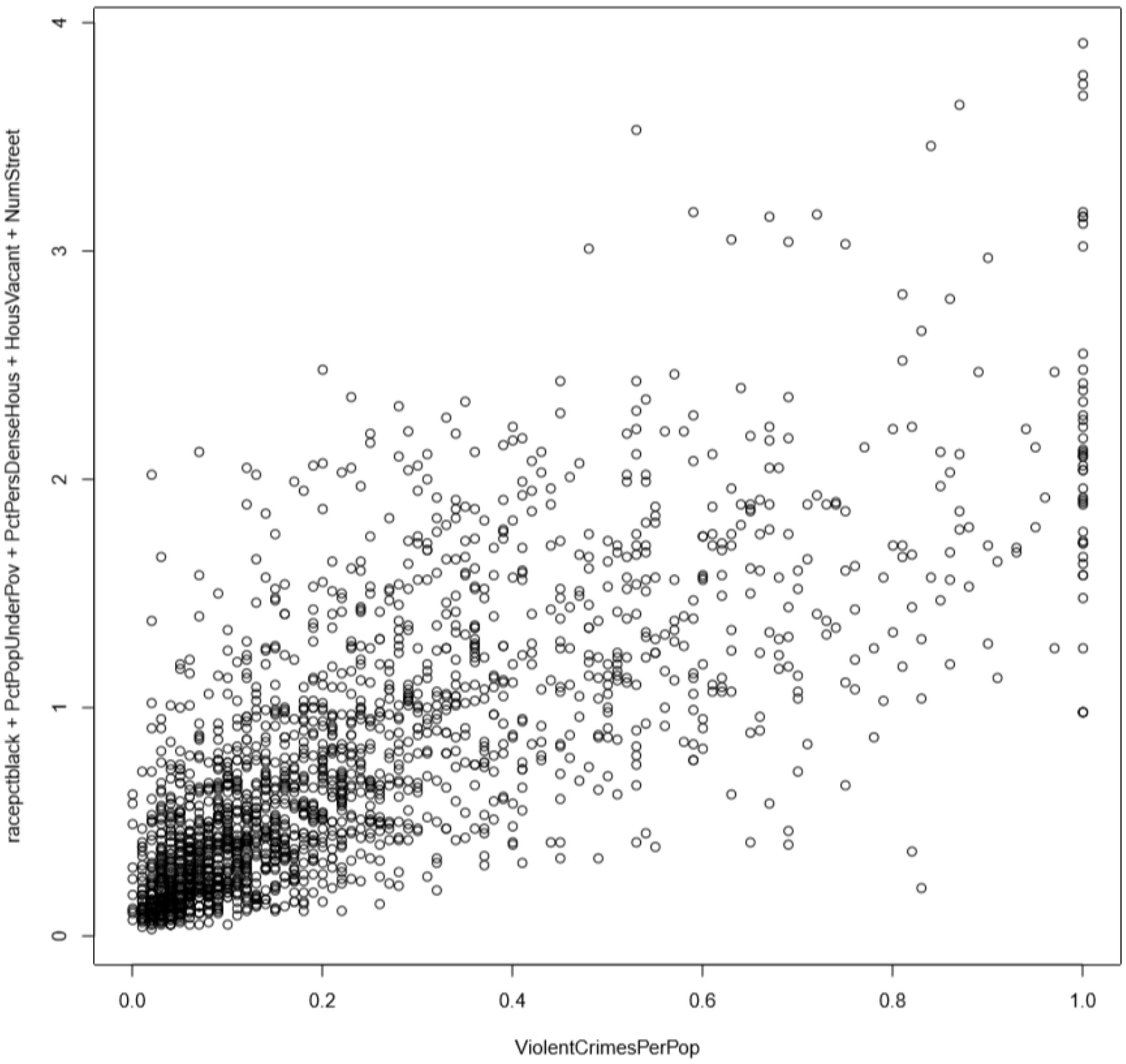
As for future studies in data science? Probabilities and statistics has been very harsh on us since day one. As for one of us who spent the most amount of time on the code, he thinks if a lenient entry-level internship opportunity were to present itself to him, he would most definitely take it, for he thinks this project was pretty fun exploring.



Top.PDF


pctWRetire + PctWorkMom + MedRent
0.5
1.0
1.5
2.0
2.5
0.0
0.2
0.4
0.6
0.8
1.0
ViolentCrimesPerPop

Middle.PDF


racepctblack + PctPopUnderPov + PctPersDenseHous + HousVacant + NumStreet
0
1
2
3
4
ViolentCrimesPerPop
0.0
0.2
0.4
0.6
0.8
1.0

Bottom.PDF