Aaron Norstrom

Brian Matamet

Ali Elsheikh

13th December 2019

Math 32 Project Final Report

Project Goal:

The primary objective of this statistical modeling project is working with somewhat large real data and analyzing it through materials covered within the course. The data analyzed in this project is with regards to communities and crime within the United States and it was collected from several different sources. Data regarding communities within the United States was collected from the United States Census Bureau in 1990. Other data sets that were used are from the Federal Bureau of Investigations in 1995 and Law Enforcement Management and Admin Stats surveys in 1990. The data set contains 1,994 rows which, each representing a different community, and 128 columns, of which 5 are non-predictive, 122 are predictive, and the last column is the real-valued response variable of which we are trying to predict.

Based on the crime data we are given, the question we are trying to answer is what variables strongly contribute to a predictive model used to calculate the number of violent crimes per 100,000 population. If we can create a model that can successfully use the data given to form an accurate prediction for this occasion, it would provide us with an important tool that could be used to predict areas where crime is developing or degrading. This question carries great significance due to the possible impact in can have in the many communities provided.

Exploring Data:

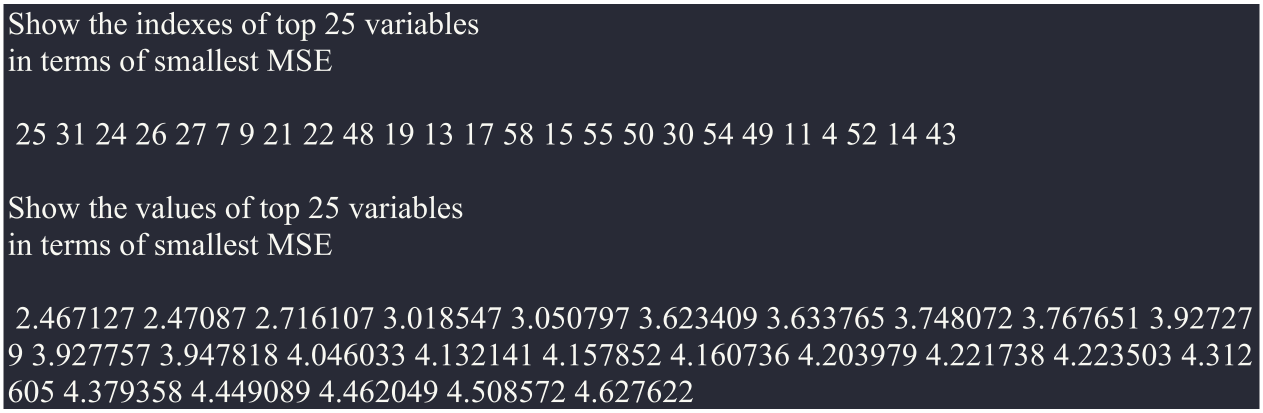
When taking notice of the different variables provided to us in the crime dataset, we found that some of the data was missing, leaving us to question if we needed to remove some variables altogether. What we found though, is that we could use a mean of the data that was provided to fill in those missing values so that we may be able to use some data we believed was vital to creating our statistical models. After resolving the issue of the missing values, we began to look for ways to condense our data so that we can best create a model with minimal error. One of the methods that we used to approach this task was to use our own intuition to select some parts of our dataset that is most likely not going to be useful to answer our question. We applied this strategy through discarding some variables from our dataset. For example, in solving our question of what the number of crimes per 100,000 population is, we are simply trying to form a prediction to the best of our ability based on the given data. For this reason, some of the predictive variables listed like the different race percentages are not going to be very helpful to the model that we are trying to build. These variables introduce a racial bias that in our model, we are attempting to avoid. We want our model to be best predictive of crime, and how it originates so that we can best find a way to target crime without it falling into a false interpretation of any certain group. Such variables include per capita income for various races, percentage of police that are a certain race, and percentage of each race for the populations.

Modeling and Methods:

For our model, we decided to go with a regression model. To sum this. We are given the values of a target variable. We call this y. For our model we are creating a variable called $\overline{y}$.

$\overline{y} = B_0 + B_1X_1 + ... + B_nX_n$ Where X_n values are our variables chosen for the model, B_n values are our intercept and coefficients for these variables, and n is the total number of variables.

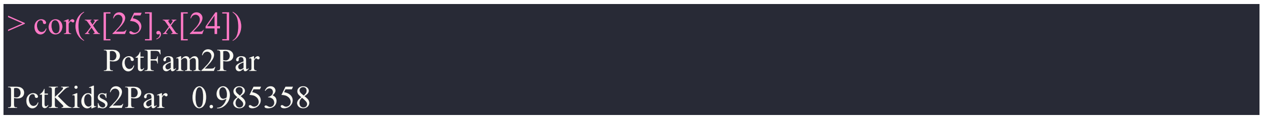
To find the best variables for our model, we made a loop in R to create a linear model with each variable and find which ones had the best mean square error (MSE) values. The mean squared error refers to finding the average of a set of errors; it is used to observe how close a regression line would be to a set of points.



```
Show the indexes of top 25 variables
in terms of smallest MSE

 25 31 24 26 27 7 9 21 22 48 19 13 17 58 15 55 50 30 54 49 11 4 52 14 43

Show the values of top 25 variables
in terms of smallest MSE

 2.467127 2.47087 2.716107 3.018547 3.050797 3.623409 3.633765 3.748072 3.767651 3.92727
9 3.927757 3.947818 4.046033 4.132141 4.157852 4.160736 4.203979 4.221738 4.223503 4.312
605 4.379358 4.449089 4.462049 4.508572 4.627622
```

Note: these MSE values are multiplied by 100 to make our data easier to read. This has no impact on the results as we only multiplied the final results.

However, in doing this we found some of the best variables are nearly the same. For the next step, we looked for variables to remove that were very similar. For example, we found two variables PctKids2Par (percentage of kids in family housing with two parents) and PctFam2Par (percentage of families (with kids) that are headed by two parents) and created a linear model for both. They ended up having almost the same exact values for estimates, t values, and error. We decided to remove PctFam2Par because it is nearly the same variable in the first place and produced nearly the same results. Which could result in overfitting the model, which we want to avoid. In class we learned that we could achieve the same results by checking the correlations of variables to each other and not just the correlation to the target variable. Some linear correlation between variables is okay and is necessary for creating a predictive linear model. However, we must be careful to not use two variables that have too high of a correlation to each other because they might represent the same thing. We had originally disregarded all negative correlations believing they would not be beneficial. However, after second thought, we found that whether a correlation is positive or negative has no effect on the predictiveness of that variable. As long as the correlation is not approaching zero or very high when compared to another variable that is not the theoretical value, we should be able to use that variable. This is because we want to avoid any conflicts introduced by multicollinearity, this problem occurs when one of our predictor variables in our multiple regression model can be linearly predicted from the others with a very high degree of accuracy. This then leads to skewed and misleading results. Therefore, for the sake of the validity, we decided to exclude a few highly correlated features. For our model we decided to not include two variables with a linear correlation coefficient with a value higher than 0.9. To do this we used the indexes of the top 25 variables that we found earlier. An example:



```
> cor(x[25],x[24])
           PctFam2Par
PctKids2Par   0.985358
```

Results:

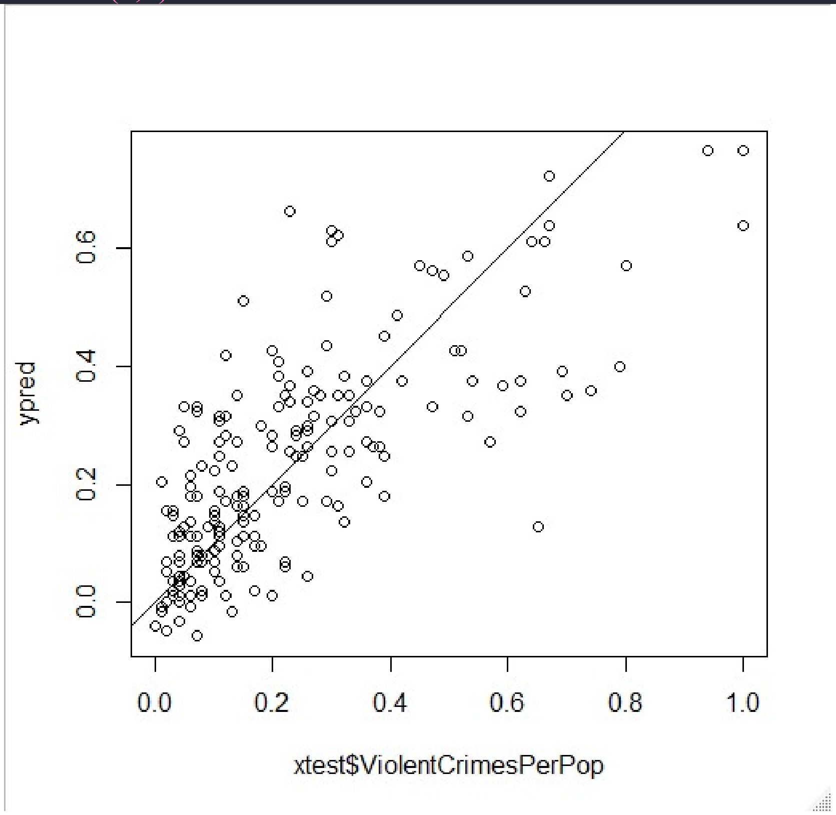
Once, deciding which variables to keep, we could start the testing. After analyzing our question that we are trying to answer, we wanted to use a heavily predictive validation technique. For this reason, we went with a cross-validation approach. Which involves splitting the data into a number of folds. We chose to do 10. From there, we could use 9 of the folds to create a model from training data and then the left-over fold would be used as test data to predict against. We looped this procedure 10 times to create 10 linear models and 10 different test data predictions. This will help to ensure our model is able to predict well over all the data. To start, we used a model that only used our number one predictor from the data set.

```
Formula used:lm(formula = ViolentCrimesPerPop ~ PctKids2Par, data = xtrain)

Average of MSE from cross-validation 2.479645
Sqrt of MSE avg or RMSE:  1.574689
```

Note: these MSE values are multiplied by 100 to make our data easier to read. This has no impact on the results as we only multiplied the final results.

To get a visual of how well our model is predicting, we plotted the predictions of the model vs the test data target variable. We added a line of best fit.

```
> plot(xtest$ViolentCrimesPerPop,ypred)
> abline(0,1)
```



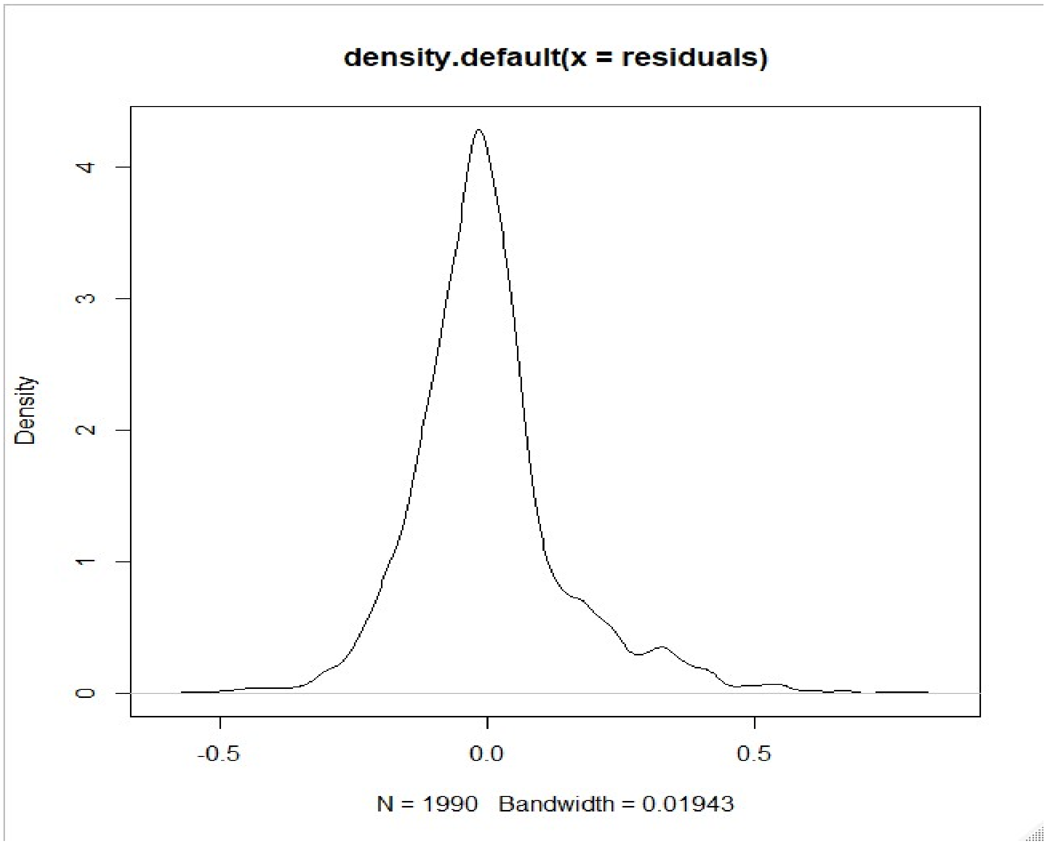
To help display our results and get an even closer look at our residuals, we used the library (fitdistrplus). This allowed to us fit our residuals into 4 different plots. A histogram of the residuals with a normal distribution density line plotted over it. A Q-Q plot. A comparison of a normal cdf and our residual cdf. And, a P-P plot. We also plotted the density of our residuals.

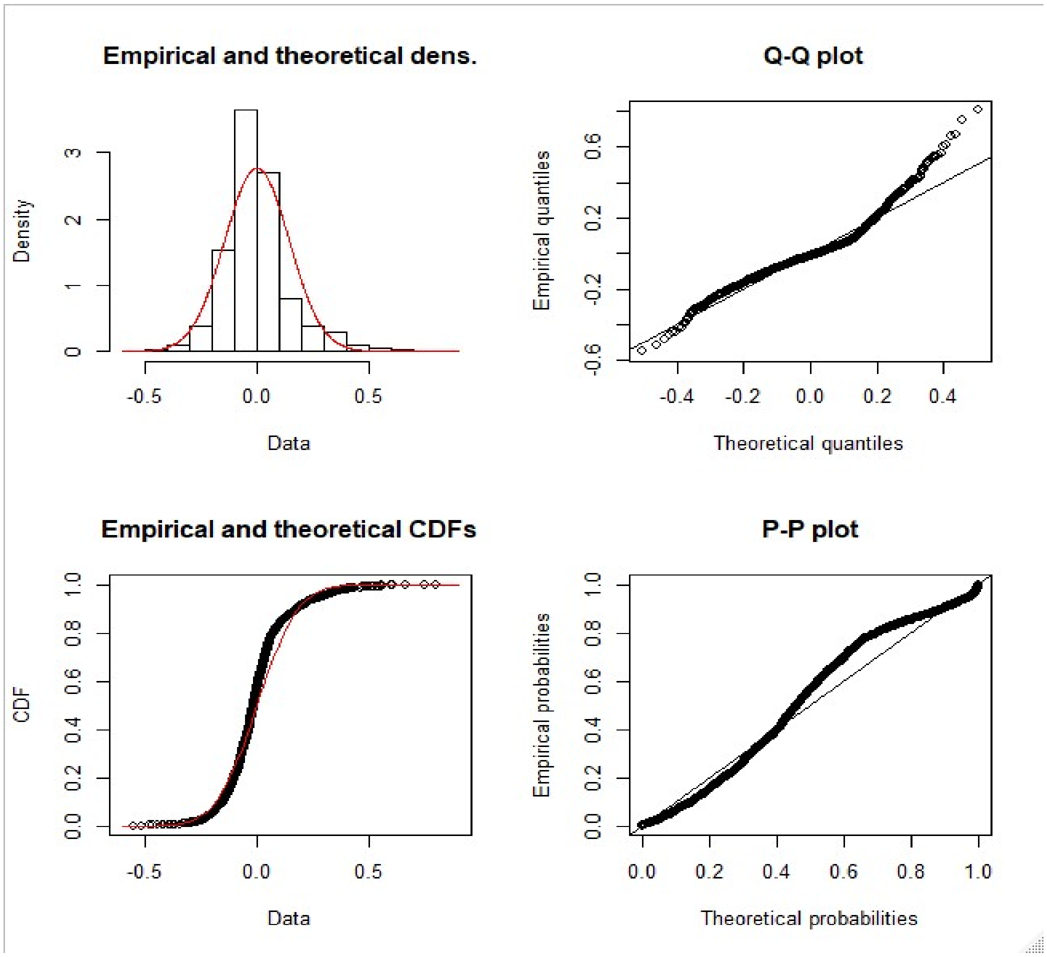
The following plot is representative of a normal Q-Q scatterplot whose sets of quantiles come from normal distributions. This scatterplot is achieved by plotting the residuals against the theoretical and the minimal deviation from the straight line indicates the normal distribution.

The empirical cdf vs theoretical cdf plot shows how close our residuals cdf is to a normal distribution cdf.

The use of the following P-P probability plot is solely for determining the degree to which 2 data sets would agree in order to evaluate the skewness of a distribution.

```
Formula used:lm(formula = ViolentCrimesPerPop ~ PctKids2Par + pctWInvInc +
    PctPersDenseHous + pctWPubAsst + PctNotHSGrad + medFamInc +
    FemalePctDiv + PctIlleg, data = xtrain)

Average of MSE from cross-validation 0.02085759
Sqrt of MSE avg or RMSE:  0.1444216
```


density.default(x = residuals)
Density
0
1
2
3
4
-0.5
0.0
0.5
N = 1990   Bandwidth = 0.01943


Empirical and theoretical dens.
Density
0
1
2
3
-0.5
0.0
0.5
Data
Q-Q plot
Empirical quantiles
-0.6
-0.2
0.2
0.6
-0.4
-0.2
0.0
0.2
0.4
Theoretical quantiles
Empirical and theoretical CDFs
CDF
0.0
0.2
0.4
0.6
0.8
1.0
-0.5
0.0
0.5
Data
P-P plot
Empirical probabilities
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Theoretical probabilities

Reflection on Process:

Over the course of this project, our main objective was to find the best predictors for crime given a sufficiently large data set. While we had some setbacks in the process of completing this project such as missing values and nearly identical variables, our team was able to get the proper guidance and was able to overcome those minor setbacks. For example, replacing the missing values for some of the variables from the data set was giving us some trouble in the start of this project which we overcame by reaching out to Professor Bhat. Some of the processes that we had done wrong was completely disregarding variables that had negative correlations. We had originally disregarded all negative correlations believing they would not be beneficial. However, after second thought, we found that whether a correlation is positive or negative has no effect on the predictiveness of that variable. As long as the correlation is not approaching zero or very high when compared to another variable that is not the theoretical value, we should be able to use that variable. This is because we want to avoid any conflicts introduced by multicollinearity, this problem occurs when one of our predictor variables in our multiple regression model can be linearly predicted from the others with a very high degree of accuracy. This then leads to skewed and misleading results. Aside from these minor setbacks and mistakes, the overall process and progress of the project went fairly well considering there were not any major complications at all. As far as other research, we looked into the realm of fantasy football. To be more specific we analyzed the possibility of building lineups from a player pool. The rules for this are simple, each player has a given salary and a given projection of points. Lineups must stay under a certain salary and fulfill every position. Our goal would be to maximize points while staying under the salary threshold, something like this could be done through optimization and proper testing.

References:

1. What Everyone Should Know about Statistical Correlation. (2017, October 5). Retrieved from https://www.americanscientist.org/article/what-everyone-should-know-about-statistical-correlation.

2. fitdistrplus: Help to Fit of a Parametric Distribution to Non-Censored or Censored Data. Retr ieved from: https://cran.r-project.org/web/packages/fitdistrplus/index.html

3. Communities within the United States. The data combines socio-economic data from the 1990 US Census, law enforcement data from the 1990 US LEMAS survey, and crime data from the 1995 FBI UCR.

Retrieved from:http://archive.ics.uci.edu/ml/datasets/communities+and+crime

The Code:

```
# Aaron Norstrom
# Ali Elsheikh
# Brian Matamet
# Math-032-Group Project-First Draft

# Our goal is to find the variables that can be used as arguments
# in our linear model to find most accurately predict violent crimes
# per population
library(fitdistrplus)
library(MASS)
# clear memory
rm(list=ls(all=TRUE))

# Load in crime data
load("crime.Rdata")

# Removing the columns of variables that we have
# decided not to use after some careful consideration
# analysis this will help remove some unneeded data loading

x <- (x[,c(6:7,17:26,34:39,44:101,103,105,107,109,116:128)])

# create a while loop to find which columns are missing data
# if column is missing entries then fill entries wiht mean
# value of that column
# This will allow us to all of the predictive variables in the set.
# Helping us to avoid removing valuable data

for(i in 1:ncol(x))
{
  if (anyNA(x[,i])){
     narows = which(is.na(x[,i]))
     x[narows,i] = mean(x[,i], na.rm = TRUE)
     }
 }

# Create a for loop to cycle through the entire
# variable list and creating a lm of the form
# lm(ViolentCrimesPerPop ~ x[,j], data = x), then
# we calculate the MSE of each one and store the
# value in our vector call mseVal

mseVal = c(ncol(x))*0
for(j in 1:ncol(x))
{
  mylm = lm(x$ViolentCrimesPerPop ~ x[,j], data = x)
  mse = mean((x$ViolentCrimesPerPop - predict(mylm))^2)
  # multiply the mse when storing to see the results better
  # this is because all the values for mse are very low due
  # to the way the data is formatted for values between 0 & 1
  mseVal[j] = mse*100
}

# Sort our vector of MSE  values for every variable
# and show the indexes of the best 25. order() shows
# the order of the indexes and sort() will show the
# actual values
order.mseVal <- order(mseVal)
sort.mseVal <- sort(mseVal)

# First value is the one being used as the first argurment
# the lm model which results in useless data and MSE, which
# is we go from 2:26 instead of 1:25

cat('\nShow the indexes of top 25 variables
in terms of smallest MSE\n\n', order.mseVal[2:26],'\n')
cat('\nShow the values of top 25 variables
in terms of smallest MSE\n\n', sort.mseVal[2:26],'\n')

# for subsets
numfolds = 10
shufflerows = sample(c(1:nrow(x)))
rowsperfold = floor(nrow(x)/ numfolds)
```

```
reshufflerows = sample(x =c(1:nrow(x)), size = nrow(x))


# Splitting data into partitions for creating test sets
# for doing cross validation
residuals = c() #vector to store our residual values from each loop
mseCross = c(1:numfolds)
 for ( i in c(1:numfolds))
 {
  si = rowsperfold*(i-1) + 1 #start index
  ei = rowsperfold*i #end index
  testrows = shufflerows[c(si:ei)]
  trainrows = shufflerows[-c(si:ei)]
  xtrain = x[trainrows,]
  xtest = x[testrows,]
  mymod = lm(ViolentCrimesPerPop ~ PctKids2Par+ pctWInvInc  + PctPersDenseHous + pctWPubAsst + PctNotHSGrad + medFamInc + FemalePctDiv + PctIlleg, data =  xtrain)
  ypred = predict(mymod, newdata = xtest)
  ytruth = (xtest$ViolentCrimesPerPop)
  residuals = c(residuals, (ytruth-ypred))
  mseCross[i] = mean((xtest$ViolentCrimesPerPop - ypred)^2)
 }


formCall = mymod$call
cat('\nFormula used:')
print(formCall)
cat('\nAverage of MSE from cross-validation', mean((mseCross)))
cat('\nSqrt of MSE avg or RMSE: ', sqrt(mean(mseCross)))
```