U.S. Crime Analysis

Justin Schulberg

5/22/2020

# U.S. Crime Analysis

In this script, I will analyze the U.S. Crime dataset included in the data folder of this repository. I will use a variety of exploratory and modeling techniques to do so, including, but not limited to:

* Outlier Detection
* Linear Regression
* Principal Component Analysis (PCA) + Lin. Reg.
* Regression Trees
* Random Forest
* Variable Selection

Let’s start by taking a look at our data:

## Warning: package 'tidyr' was built under R version 3.6.3

## Warning: package 'factoextra' was built under R version 3.6.2

## Warning: package 'here' was built under R version 3.6.3

## Warning: package 'rpart.plot' was built under R version 3.6.2

## Warning: package 'glmnet' was built under R version 3.6.3

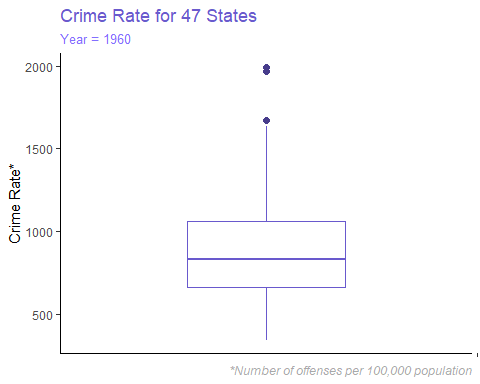
## # A tibble: 6 x 16  
## M So Ed Po1 Po2 LF M.F Pop NW U1 U2 Wealth Ineq  
## <dbl> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int> <dbl>  
## 1 15.1 1 9.1 5.8 5.6 0.51 95 33 30.1 0.108 4.1 3940 26.1  
## 2 14.3 0 11.3 10.3 9.5 0.583 101. 13 10.2 0.096 3.6 5570 19.4  
## 3 14.2 1 8.9 4.5 4.4 0.533 96.9 18 21.9 0.094 3.3 3180 25   
## 4 13.6 0 12.1 14.9 14.1 0.577 99.4 157 8 0.102 3.9 6730 16.7  
## 5 14.1 0 12.1 10.9 10.1 0.591 98.5 18 3 0.091 2 5780 17.4  
## 6 12.1 0 11 11.8 11.5 0.547 96.4 25 4.4 0.084 2.9 6890 12.6  
## # ... with 3 more variables: Prob <dbl>, Time <dbl>, Crime <int>

## M So Ed Po1   
## Min. :11.90 Min. :0.0000 Min. : 8.70 Min. : 4.50   
## 1st Qu.:13.00 1st Qu.:0.0000 1st Qu.: 9.75 1st Qu.: 6.25   
## Median :13.60 Median :0.0000 Median :10.80 Median : 7.80   
## Mean :13.86 Mean :0.3404 Mean :10.56 Mean : 8.50   
## 3rd Qu.:14.60 3rd Qu.:1.0000 3rd Qu.:11.45 3rd Qu.:10.45   
## Max. :17.70 Max. :1.0000 Max. :12.20 Max. :16.60   
## Po2 LF M.F Pop   
## Min. : 4.100 Min. :0.4800 Min. : 93.40 Min. : 3.00   
## 1st Qu.: 5.850 1st Qu.:0.5305 1st Qu.: 96.45 1st Qu.: 10.00   
## Median : 7.300 Median :0.5600 Median : 97.70 Median : 25.00   
## Mean : 8.023 Mean :0.5612 Mean : 98.30 Mean : 36.62   
## 3rd Qu.: 9.700 3rd Qu.:0.5930 3rd Qu.: 99.20 3rd Qu.: 41.50   
## Max. :15.700 Max. :0.6410 Max. :107.10 Max. :168.00   
## NW U1 U2 Wealth   
## Min. : 0.20 Min. :0.07000 Min. :2.000 Min. :2880   
## 1st Qu.: 2.40 1st Qu.:0.08050 1st Qu.:2.750 1st Qu.:4595   
## Median : 7.60 Median :0.09200 Median :3.400 Median :5370   
## Mean :10.11 Mean :0.09547 Mean :3.398 Mean :5254   
## 3rd Qu.:13.25 3rd Qu.:0.10400 3rd Qu.:3.850 3rd Qu.:5915   
## Max. :42.30 Max. :0.14200 Max. :5.800 Max. :6890   
## Ineq Prob Time Crime   
## Min. :12.60 Min. :0.00690 Min. :12.20 Min. : 342.0   
## 1st Qu.:16.55 1st Qu.:0.03270 1st Qu.:21.60 1st Qu.: 658.5   
## Median :17.60 Median :0.04210 Median :25.80 Median : 831.0   
## Mean :19.40 Mean :0.04709 Mean :26.60 Mean : 905.1   
## 3rd Qu.:22.75 3rd Qu.:0.05445 3rd Qu.:30.45 3rd Qu.:1057.5   
## Max. :27.60 Max. :0.11980 Max. :44.00 Max. :1993.0

Below is a list of the variables in data along with their associated descriptions. We want to predict the last column, Crime, based on the other predictor variables.  
Variable Description  
M percentage of males aged 14-24 in total state population  
So indicator variable for a southern state  
Ed mean years of schooling of the population aged 25 years or over  
Po1 per capita expenditure on police protection in 1960  
Po2 per capita expenditure on police protection in 1959  
LF labour force participation rate of civilian urban males in the age-group 14-24  
M.F number of males per 100 females  
Pop state population in 1960 in hundred thousands  
NW percentage of nonwhites in the population  
U1 unemployment rate of urban males 14-24  
U2 unemployment rate of urban males 35-39  
Wealth wealth: median value of transferable assets or family income  
Ineq income inequality: percentage of families earning below half the median income  
Prob probability of imprisonment: ratio of number of commitments to number of offenses  
Time average time in months served by offenders in state prisons before their first release  
Crime crime rate: number of offenses per 100,000 population in 1960

# Outliers

In this section, we’ll first test to see whether there are any outliers in the last column (number of crimes per 100,000 people). To do so, we’ll use the grubbs.test function in the outliers package in R.



Now that we have a clear idea of what the outliers could look like, let’s use the grubbs.test to better understand these outliers.

##   
## Grubbs test for one outlier  
##   
## data: data\_grubbs$Crime  
## G = 2.81287, U = 0.82426, p-value = 0.07887  
## alternative hypothesis: highest value 1993 is an outlier

## [1] "highest value 1993 is an outlier"

## [1] 0.07887486

Since the p-value is above .05 (.079, to be exact), we cannot say with confidence that there is an outlier in the set. Let’s move on to some modelling.

# Linear Regression

In the first section, we’ll use a simple linear regression model to predict on our crime data. Start by scaling the data so it’s standardardized. Avoid column 2 because it’s an indicator (factor) for southern states.

##   
## Call:  
## lm(formula = Crime ~ ., data = data\_scaled)  
##   
## Coefficients:  
## (Intercept) M Ed Po1 Po2 LF   
## 0.003348 0.285399 0.544723 1.481515 -0.791075 -0.069361   
## M.F Pop NW U1 U2 Wealth   
## 0.132622 -0.072154 0.111784 -0.271628 0.366412 0.239919   
## Ineq Prob Time So   
## 0.729010 -0.285431 -0.063748 -0.009834

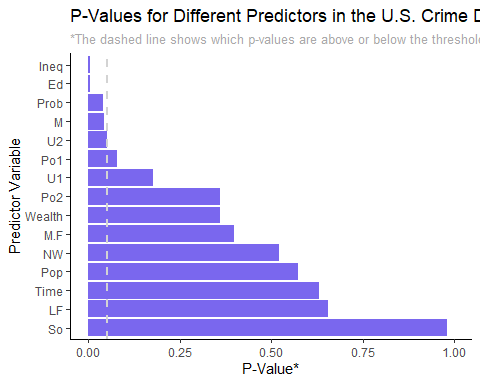
##   
## Call:  
## lm(formula = Crime ~ ., data = data\_scaled)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.02321 -0.25361 -0.01731 0.29214 1.32554   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.003348 0.152841 0.022 0.98267   
## M 0.285399 0.135547 2.106 0.04344 \*   
## Ed 0.544723 0.179589 3.033 0.00486 \*\*  
## Po1 1.481515 0.815350 1.817 0.07889 .   
## Po2 -0.791075 0.849313 -0.931 0.35883   
## LF -0.069361 0.153568 -0.452 0.65465   
## M.F 0.132622 0.155075 0.855 0.39900   
## Pop -0.072154 0.126938 -0.568 0.57385   
## NW 0.111784 0.172308 0.649 0.52128   
## U1 -0.271628 0.196261 -1.384 0.17624   
## U2 0.366412 0.179791 2.038 0.05016 .   
## Wealth 0.239919 0.258630 0.928 0.36075   
## Ineq 0.729010 0.234330 3.111 0.00398 \*\*  
## Prob -0.285431 0.133588 -2.137 0.04063 \*   
## Time -0.063748 0.131294 -0.486 0.63071   
## So -0.009834 0.384616 -0.026 0.97977   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5405 on 31 degrees of freedom  
## Multiple R-squared: 0.8031, Adjusted R-squared: 0.7078   
## F-statistic: 8.429 on 15 and 31 DF, p-value: 0.0000003539

The first two items we should look at are the R-squared value (80.3%) and the Adjusted R-squared value (71.8%). The next item we should look at is the overal p-value, which is p-value: 0.0000003539. This means that our crime data can be explained by the amalgam of predictor variables we provided.

## (Intercept) M Ed Po1 Po2 LF   
## 0.003347768 0.285399152 0.544722603 1.481514913 -0.791074563 -0.069361443   
## M.F Pop NW U1 U2 Wealth   
## 0.132622452 -0.072154040 0.111784244 -0.271627975 0.366411688 0.239918991   
## Ineq Prob Time So   
## 0.729009903 -0.285430950 -0.063748223 -0.009834067

As an example of what this means, if M (percentage of males aged 14-24 in total state population) increases by 1, we would expect crime to increase by 87.83 crimes per 100,000 population.

But which of the variables mattered and which did not? Well, to start, let’s look at the individual p-values for each variable we have. We’ll do this by pulling out the p-values and coefficient names.



Let’s confirm what we see above, especially since U2 seems on the border. Filter our data frame to only show p-values less than .05 and then order our results.

Now that we have the coefficients that matter, let’s re-run the linear model only on the coefficients we see above

##   
## Call:  
## lm(formula = Crime ~ ., data = data\_signif)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.3780 -0.6568 -0.1441 0.3563 2.4827   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.0000000000000000604 0.1310586847078026862 0.000 1.00000   
## M 0.1168920934073441331 0.1735000214758070369 0.674 0.50417   
## Ed 0.4298365505224954752 0.2080133416220263098 2.066 0.04499 \*   
## Ineq 0.2772215506837938381 0.2349037199509093621 1.180 0.24458   
## Prob -0.4310280070497272686 0.1505129457305220686 -2.864 0.00651 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8985 on 42 degrees of freedom  
## Multiple R-squared: 0.2629, Adjusted R-squared: 0.1927   
## F-statistic: 3.745 on 4 and 42 DF, p-value: 0.01077

We see from this model that our R-squared value dropped significantly. Very strange, and not what I expected. Let’s try bringing in the two variables that were just above the .05 threshold (U@ and Po1).

##   
## Call:  
## lm(formula = Crime ~ ., data = data\_updated)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.3658 -0.1913 -0.0181 0.3614 1.3014   
##   
## Coefficients:  
## Estimate Std. Error t value  
## (Intercept) -0.0000000000000001558 0.0790947003719782299 0.000  
## M 0.2589438160798205324 0.1059969678013346767 2.443  
## Ed 0.4632372705133553925 0.1255970724844087949 3.688  
## Ineq 0.7046465259134735426 0.1501861745051313868 4.692  
## Prob -0.2273487874016537624 0.0938579544450350439 -2.422  
## Po1 0.9315281453651754751 0.1080581397067851696 8.621  
## Pr(>|t|)   
## (Intercept) 1.000000   
## M 0.018964 \*   
## Ed 0.000656 \*\*\*  
## Ineq 0.0000300483391 \*\*\*  
## Prob 0.019930 \*   
## Po1 0.0000000000947 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5422 on 41 degrees of freedom  
## Multiple R-squared: 0.7379, Adjusted R-squared: 0.706   
## F-statistic: 23.09 on 5 and 41 DF, p-value: 0.00000000005926

Great! Our R-squared value went up to 76.6% and our adjusted R-squared went up to 73.1%. In particular, we notice that our R-squared value has gone down (it was originally 80.3%); however, our adjusted R-squared value has gone up (it was originally 70.1%). This makes sense because adjusted R-squared factors in the number of variables, and should get closer to 1 the less we overfit our model.

Now, let’s predict on a fake dataset given by the original prompt to see what we get.

## (Intercept) M Ed   
## -0.000000000000000155812 0.258943816079820532394 0.463237270513355392509   
## Ineq Prob Po1   
## 0.704646525913473542602 -0.227348787401653762430 0.931528145365175475057

## 1   
## 33.59023

The linear regression model estimates that the Crime rate based on this data will be 35.79. This translates to about 36 crimes per 100,000 population. This seems extremely off from our original dataset, so we’ll instead resort to our original, unscaled data which included every variable.

Now let’s take a look at the confidence interval for this new data.

## fit lwr upr  
## 1 33.59023 24.58019 42.60027

The same linear regression model estimates 798 crimes per 100,000 population. It also estimates that the lower limit for our data is -886.6 and the upper limit is 2482.3. This result makes more sense based on the fact that the median of our original crime is **831**.

# Principal Component Analysis

In this section, I’ll apply Principal Component Analysis, a method for trimming down the number of variables in our dataset and for variable selection. This *should* help with model-building later on.

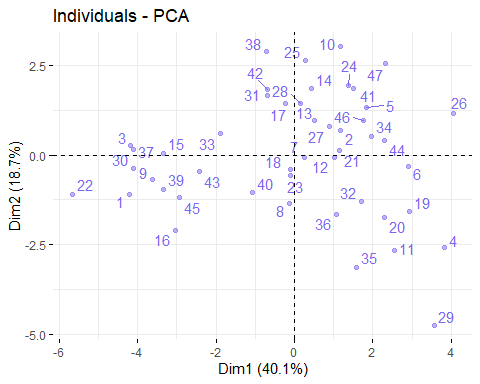
## # A tibble: 6 x 15  
## M So Ed Po1 Po2 LF M.F Pop NW U1 U2 Wealth Ineq  
## <dbl> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int> <dbl>  
## 1 15.1 1 9.1 5.8 5.6 0.51 95 33 30.1 0.108 4.1 3940 26.1  
## 2 14.3 0 11.3 10.3 9.5 0.583 101. 13 10.2 0.096 3.6 5570 19.4  
## 3 14.2 1 8.9 4.5 4.4 0.533 96.9 18 21.9 0.094 3.3 3180 25   
## 4 13.6 0 12.1 14.9 14.1 0.577 99.4 157 8 0.102 3.9 6730 16.7  
## 5 14.1 0 12.1 10.9 10.1 0.591 98.5 18 3 0.091 2 5780 17.4  
## 6 12.1 0 11 11.8 11.5 0.547 96.4 25 4.4 0.084 2.9 6890 12.6  
## # ... with 2 more variables: Prob <dbl>, Time <dbl>

Run our first Principle Component Analysis using the prcomp() function which uses Singular value decomposition (SVD), which examines the covariances / correlations between individual observations.

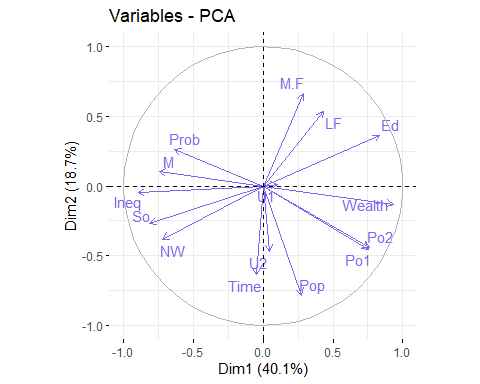
## Standard deviations (1, .., p=15):  
## [1] 2.45335539 1.67387187 1.41596057 1.07805742 0.97892746 0.74377006  
## [7] 0.56729065 0.55443780 0.48492813 0.44708045 0.41914843 0.35803646  
## [13] 0.26332811 0.24180109 0.06792764  
##   
## Rotation (n x k) = (15 x 15):  
## PC1 PC2 PC3 PC4 PC5  
## M -0.30371194 0.06280357 0.1724199946 -0.02035537 -0.35832737  
## So -0.33088129 -0.15837219 0.0155433104 0.29247181 -0.12061130  
## Ed 0.33962148 0.21461152 0.0677396249 0.07974375 -0.02442839  
## Po1 0.30863412 -0.26981761 0.0506458161 0.33325059 -0.23527680  
## Po2 0.31099285 -0.26396300 0.0530651173 0.35192809 -0.20473383  
## LF 0.17617757 0.31943042 0.2715301768 -0.14326529 -0.39407588  
## M.F 0.11638221 0.39434428 -0.2031621598 0.01048029 -0.57877443  
## Pop 0.11307836 -0.46723456 0.0770210971 -0.03210513 -0.08317034  
## NW -0.29358647 -0.22801119 0.0788156621 0.23925971 -0.36079387  
## U1 0.04050137 0.00807439 -0.6590290980 -0.18279096 -0.13136873  
## U2 0.01812228 -0.27971336 -0.5785006293 -0.06889312 -0.13499487  
## Wealth 0.37970331 -0.07718862 0.0100647664 0.11781752 0.01167683  
## Ineq -0.36579778 -0.02752240 -0.0002944563 -0.08066612 -0.21672823  
## Prob -0.25888661 0.15831708 -0.1176726436 0.49303389 0.16562829  
## Time -0.02062867 -0.38014836 0.2235664632 -0.54059002 -0.14764767  
## PC6 PC7 PC8 PC9 PC10 PC11  
## M -0.449132706 -0.15707378 -0.55367691 0.15474793 -0.01443093 0.39446657  
## So -0.100500743 0.19649727 0.22734157 -0.65599872 0.06141452 0.23397868  
## Ed -0.008571367 -0.23943629 -0.14644678 -0.44326978 0.51887452 -0.11821954  
## Po1 -0.095776709 0.08011735 0.04613156 0.19425472 -0.14320978 -0.13042001  
## Po2 -0.119524780 0.09518288 0.03168720 0.19512072 -0.05929780 -0.13885912  
## LF 0.504234275 -0.15931612 0.25513777 0.14393498 0.03077073 0.38532827  
## M.F -0.074501901 0.15548197 -0.05507254 -0.24378252 -0.35323357 -0.28029732  
## Pop 0.547098563 0.09046187 -0.59078221 -0.20244830 -0.03970718 0.05849643  
## NW 0.051219538 -0.31154195 0.20432828 0.18984178 0.49201966 -0.20695666  
## U1 0.017385981 -0.17354115 -0.20206312 0.02069349 0.22765278 -0.17857891  
## U2 0.048155286 -0.07526787 0.24369650 0.05576010 -0.04750100 0.47021842  
## Wealth -0.154683104 -0.14859424 0.08630649 -0.23196695 -0.11219383 0.31955631  
## Ineq 0.272027031 0.37483032 0.07184018 -0.02494384 -0.01390576 -0.18278697  
## Prob 0.283535996 -0.56159383 -0.08598908 -0.05306898 -0.42530006 -0.08978385  
## Time -0.148203050 -0.44199877 0.19507812 -0.23551363 -0.29264326 -0.26363121  
## PC12 PC13 PC14 PC15  
## M 0.16580189 -0.05142365 0.04901705 0.0051398012  
## So -0.05753357 -0.29368483 -0.29364512 0.0084369230  
## Ed 0.47786536 0.19441949 0.03964277 -0.0280052040  
## Po1 0.22611207 -0.18592255 -0.09490151 -0.6894155129  
## Po2 0.19088461 -0.13454940 -0.08259642 0.7200270100  
## LF 0.02705134 -0.27742957 -0.15385625 0.0336823193  
## M.F -0.23925913 0.31624667 -0.04125321 0.0097922075  
## Pop -0.18350385 0.12651689 -0.05326383 0.0001496323  
## NW -0.36671707 0.22901695 0.13227774 -0.0370783671  
## U1 -0.09314897 -0.59039450 -0.02335942 0.0111359325  
## U2 0.28440496 0.43292853 -0.03985736 0.0073618948  
## Wealth -0.32172821 -0.14077972 0.70031840 -0.0025685109  
## Ineq 0.43762828 -0.12181090 0.59279037 0.0177570357  
## Prob 0.15567100 -0.03547596 0.04761011 0.0293376260  
## Time 0.13536989 -0.05738113 -0.04488401 0.0376754405

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## Standard deviation 2.4534 1.6739 1.4160 1.07806 0.97893 0.74377 0.56729  
## Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688 0.02145  
## Cumulative Proportion 0.4013 0.5880 0.7217 0.79920 0.86308 0.89996 0.92142  
## PC8 PC9 PC10 PC11 PC12 PC13 PC14  
## Standard deviation 0.55444 0.48493 0.44708 0.41915 0.35804 0.26333 0.2418  
## Proportion of Variance 0.02049 0.01568 0.01333 0.01171 0.00855 0.00462 0.0039  
## Cumulative Proportion 0.94191 0.95759 0.97091 0.98263 0.99117 0.99579 0.9997  
## PC15  
## Standard deviation 0.06793  
## Proportion of Variance 0.00031  
## Cumulative Proportion 1.00000

We can see that, from our Cumulative Proportion dimension, the first six principle components explain about 90% of the variation in the data.



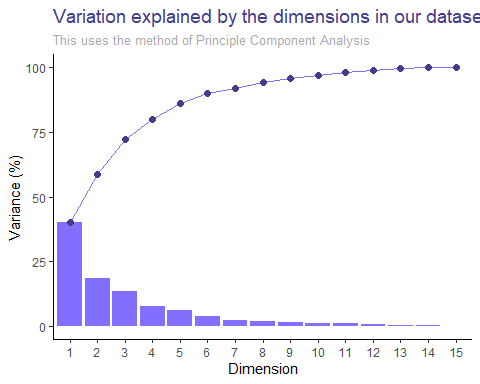
Above is a graph of the individual observations. Individual states with a similar profile of factors will be grouped. Next, we’ll create a graph of the variables (columns in our dataset).  
- Positively correlated variables point to the same side of the plot.  
- Negatively correlated variables point to opposite sides of the plot.



What are our eigenvalues?

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 6.018952657 40.1263510 40.12635  
## Dim.2 2.801847026 18.6789802 58.80533  
## Dim.3 2.004944334 13.3662956 72.17163  
## Dim.4 1.162207801 7.7480520 79.91968  
## Dim.5 0.958298972 6.3886598 86.30834  
## Dim.6 0.553193900 3.6879593 89.99630  
## Dim.7 0.321818687 2.1454579 92.14176  
## Dim.8 0.307401270 2.0493418 94.19110  
## Dim.9 0.235155292 1.5677019 95.75880  
## Dim.10 0.199880931 1.3325395 97.09134  
## Dim.11 0.175685403 1.1712360 98.26258  
## Dim.12 0.128190107 0.8546007 99.11718  
## Dim.13 0.069341691 0.4622779 99.57945  
## Dim.14 0.058467765 0.3897851 99.96924  
## Dim.15 0.004614165 0.0307611 100.00000

Note that the variance.percent values noted here align with our screeplot above. This also now tells us that 90% of the variance in our data is explained by just 6 dimensions. Let’s graph the cumulative variance percentages so it’s easier to visualize.



Just like earlier, we’ll bring our crime data (our dependent variable) back in and see how our predictions are using the principle components.

##   
## Call:  
## lm(formula = Crime ~ ., data = pca\_vals)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -377.15 -172.23 25.81 132.10 480.38   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 905.09 35.35 25.604 < 0.0000000000000002 \*\*\*  
## PC1 65.22 14.56 4.478 0.000061443 \*\*\*  
## PC2 -70.08 21.35 -3.283 0.00214 \*\*   
## PC3 25.19 25.23 0.998 0.32409   
## PC4 69.45 33.14 2.095 0.04252 \*   
## PC5 -229.04 36.50 -6.275 0.000000194 \*\*\*  
## PC6 -60.21 48.04 -1.253 0.21734   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 242.3 on 40 degrees of freedom  
## Multiple R-squared: 0.6586, Adjusted R-squared: 0.6074   
## F-statistic: 12.86 on 6 and 40 DF, p-value: 0.00000004869

Notice that our adjusted R-squared value is 60.7%. This is significantly lower than what we saw in the previous homework (somewhere around 75-80%), but we’ve significantly reduced the number of variables we’re using.

With that in hand, we’ll transform our data back, since PCA performs a linear transformation to our dataset.

## [1] "Great work. Let's keep on going."

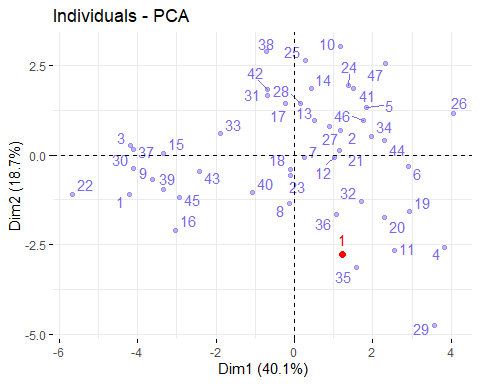
## V1   
## Min. : 273.7   
## 1st Qu.: 705.3   
## Median : 884.0   
## Mean : 905.1   
## 3rd Qu.:1102.2   
## Max. :1874.5

The estimates represent what the PCA model expects our crime data to be based on the six principle components we identified. These are not exact, because our PCA model really only explains about 90% of the variation within our dataset.

As we did earlier, now, let’s predict on the dataset we’ve been provided. First we’ll store all the new data we’ve been provided into a data frame.

## PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8  
## 1 1.224044 -2.767641 0.533605 -1.146837 -1.206098 2.333343 -0.1535916 -1.391625  
## PC9 PC10 PC11 PC12 PC13 PC14 PC15  
## 1 1.460274 -0.4525158 -0.3466498 1.663782 -1.811307 -2.174071 1.288675

So where does this new “state” fit? We’ll use the graph we saw earlier and add this new point, manipulated utilizing our PCA, as a red dot on the graph.



Now we’ll predict what the crime rate would now be by leveraging the new data that’s had PCA applied.

## 1   
## 1248.427

Thus we predict that the crime rate in the new state would be 1248 crimes per 100,000 population. This seems within the range of reason for our data, especially given that the mean of our crime data is **905.0851064** and the range goes from **342** to **1993**.

# Decision Trees

In this section, I’ll find the best model available using: - Regression Trees - Random Forest

## Regression Tree

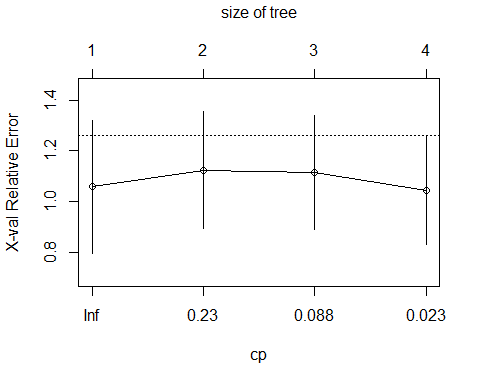
Recursive partitioning is a fundamental tool in data mining. It helps us explore the stucture of a set of data, while developing easy to visualize decision rules for predicting a categorical (classification tree) or continuous (regression tree) outcome. Our formula below will be in the format outcome ~ ., which allows us to predict on all of our variables.

We’ll grow our tree using the rpart function from the rpart package to create our regression tree.

## Call:  
## rpart(formula = Crime ~ ., data = data\_scaled\_fix, method = "anova")  
## n= 47   
##   
## CP nsplit rel error xerror xstd  
## 1 0.36296293 0 1.0000000 1.059023 0.2612971  
## 2 0.14814320 1 0.6370371 1.123717 0.2301792  
## 3 0.05173165 2 0.4888939 1.113252 0.2247664  
## 4 0.01000000 3 0.4371622 1.045636 0.2140906  
##   
## Variable importance  
## Po1 Po2 Wealth Ineq Prob M NW Pop Time Ed LF   
## 18 17 11 11 10 10 9 5 4 4 1   
##   
## Node number 1: 47 observations, complexity param=0.3629629  
## mean=-1.039358e-16, MSE=0.9787234   
## left son=2 (23 obs) right son=3 (24 obs)  
## Primary splits:  
## Po1 < -0.2860126 to the left, improve=0.3629629, (0 missing)  
## Po2 < -0.2944798 to the left, improve=0.3629629, (0 missing)  
## Prob < -0.2305884 to the right, improve=0.3217700, (0 missing)  
## NW < -0.2395015 to the left, improve=0.2356621, (0 missing)  
## Wealth < 1.022034 to the left, improve=0.2002403, (0 missing)  
## Surrogate splits:  
## Po2 < -0.2944798 to the left, agree=1.000, adj=1.000, (0 split)  
## Wealth < 0.07894027 to the left, agree=0.830, adj=0.652, (0 split)  
## Prob < -0.1536433 to the right, agree=0.809, adj=0.609, (0 split)  
## M < -0.4833422 to the right, agree=0.745, adj=0.478, (0 split)  
## Ineq < -0.5639655 to the right, agree=0.745, adj=0.478, (0 split)  
##   
## Node number 2: 23 observations, complexity param=0.05173165  
## mean=-0.6088395, MSE=0.2264937   
## left son=4 (12 obs) right son=5 (11 obs)  
## Primary splits:  
## Pop < -0.3708059 to the left, improve=0.4568043, (0 missing)  
## M < 0.5112762 to the left, improve=0.3931567, (0 missing)  
## NW < -0.4583118 to the left, improve=0.3184074, (0 missing)  
## Po1 < -0.9253348 to the left, improve=0.2310098, (0 missing)  
## U1 < -0.1368969 to the right, improve=0.2119062, (0 missing)  
## Surrogate splits:  
## NW < -0.4583118 to the left, agree=0.826, adj=0.636, (0 split)  
## M < 0.5112762 to the left, agree=0.783, adj=0.545, (0 split)  
## Time < -0.6063828 to the left, agree=0.783, adj=0.545, (0 split)  
## Ed < 0.2558061 to the right, agree=0.739, adj=0.455, (0 split)  
## Po1 < -0.9589833 to the left, agree=0.739, adj=0.455, (0 split)  
##   
## Node number 3: 24 observations, complexity param=0.1481432  
## mean=0.5834712, MSE=1.003931   
## left son=6 (10 obs) right son=7 (14 obs)  
## Primary splits:  
## NW < -0.2395015 to the left, improve=0.2828293, (0 missing)  
## M < -0.6424812 to the left, improve=0.2714159, (0 missing)  
## Time < -0.6628885 to the left, improve=0.2060170, (0 missing)  
## M.F < 0.3047006 to the left, improve=0.1703438, (0 missing)  
## Po2 < 0.6174944 to the left, improve=0.1659433, (0 missing)  
## Surrogate splits:  
## Ed < 0.792143 to the right, agree=0.750, adj=0.4, (0 split)  
## Ineq < -0.7895516 to the left, agree=0.750, adj=0.4, (0 split)  
## Time < -0.6628885 to the left, agree=0.750, adj=0.4, (0 split)  
## Pop < -0.1738065 to the left, agree=0.708, adj=0.3, (0 split)  
## LF < 0.6757556 to the right, agree=0.667, adj=0.2, (0 split)  
##   
## Node number 4: 12 observations  
## mean=-0.9168028, MSE=0.135826   
##   
## Node number 5: 11 observations  
## mean=-0.2728796, MSE=0.1090716   
##   
## Node number 6: 10 observations  
## mean=-0.04701877, MSE=0.3727469   
##   
## Node number 7: 14 observations  
## mean=1.033821, MSE=0.9680209

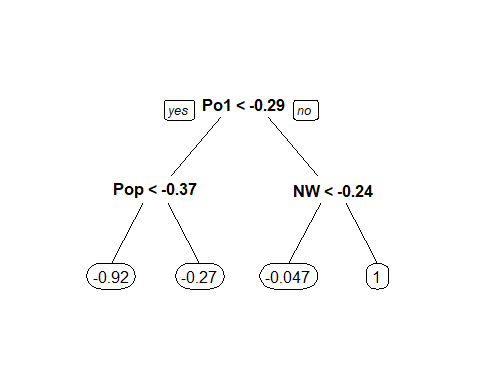
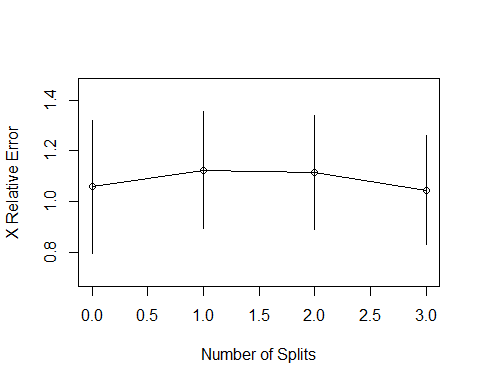
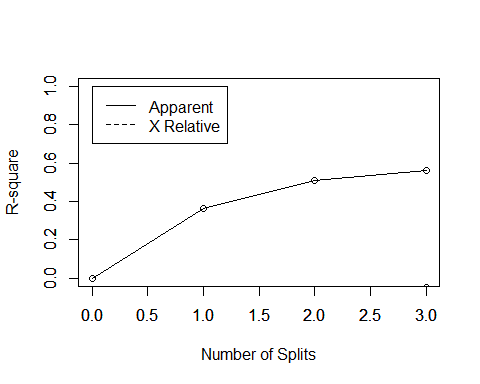
Immediately, we see that our four most important variables in this tree model are:  
1. Po1 (per capita expenditure on police protection in 1960)  
2. Po2 (per capita expenditure on police protection in 1959)  
3. Wealth (median value of transferable assets or family income)  
4. Ineq (percentage of families earning below half the median income)

## CP nsplit rel error xerror xstd  
## 1 0.36296293 0 1.0000000 1.059023 0.2612971  
## 2 0.14814320 1 0.6370371 1.123717 0.2301792  
## 3 0.05173165 2 0.4888939 1.113252 0.2247664  
## 4 0.01000000 3 0.4371622 1.045636 0.2140906



Notice from the graph of the relative errors, that our best split occurs when cp = 4, which is where the lowest cross valiadation error occurs. This means that the optimal number of splits for our tree is 4. This value occurs at cp = **0.01**.

##   
## Regression tree:  
## rpart(formula = Crime ~ ., data = data\_scaled\_fix, method = "anova")  
##   
## Variables actually used in tree construction:  
## [1] NW Po1 Pop  
##   
## Root node error: 46/47 = 0.97872  
##   
## n= 47   
##   
## CP nsplit rel error xerror xstd  
## 1 0.362963 0 1.00000 1.0590 0.26130  
## 2 0.148143 1 0.63704 1.1237 0.23018  
## 3 0.051732 2 0.48889 1.1133 0.22477  
## 4 0.010000 3 0.43716 1.0456 0.21409



From the visualization of the regression tree above, we can see that Po1 is the most significant variable (which aligns with the output of our rpart() function), and then Pop and NW are the next most important. After that, not many of the variables are that important for our purposes.

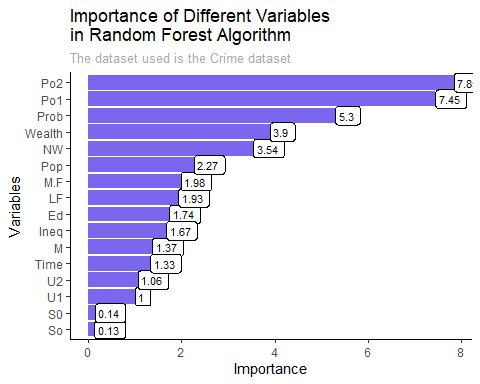
## Random Forest

Now we’ll compare our results to those we get from running a Random Forest model. The Random Forest model will make 500 decision trees and “vote” on the nodes that work best (i.e. provide the highest accuracy).

##   
## Call:  
## randomForest(formula = Crime ~ ., data = data\_scaled\_fix)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 5  
##   
## Mean of squared residuals: 0.570131  
## % Var explained: 41.75

Here we note that the mean of the squared residuals is **0.570131**. The Random Forest algorithm we ran explains about 56.7% of the variation in our data, which is not great.

Which variables were the most important according to the Random Forest algorithm?



It’s interesting to note here that the three most important variables for the Random Forest algorithm are similar to the three we found in the Regression Tree earlier:  
1. Po1 (per capita expenditure on police protection in 1960)  
2. Po2 (per capita expenditure on police protection in 1959)  
3. Wealth (median value of transferable assets or family income)

# Variable Selection

In this section, I’ll attempt multiple methods for narrowing down our dataset by selecting the most important variables. As I narrow down the dataset, I’ll compare different linear regression models to see if our accuracy improves.

## Stepwise Regression

First we’ll find the full linear regression model and then run Stepwise Regression, in which we’ll slowly pick apart the full model by slicing out negligible coefficients.

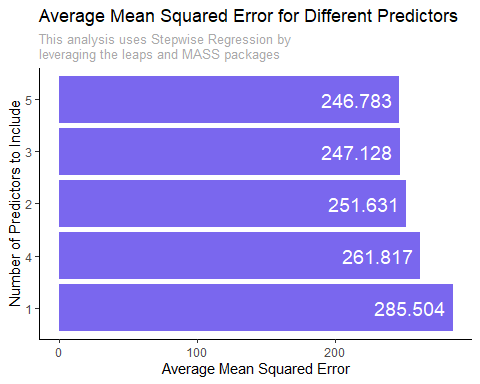
##   
## Call:  
## lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,   
## data = data\_scaled)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.14981 -0.28718 0.00784 0.31581 1.24960   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -0.0000000000000003591 0.0737492532532140616 0.000 1.00000  
## M 0.3032430653453815350 0.1088472638915121971 2.786 0.00828  
## Ed 0.5209921876198559954 0.1525872689769799673 3.414 0.00153  
## Po1 0.7887902674851503537 0.1192860516465484993 6.613 0.0000000826  
## M.F 0.1702060389736743118 0.1036268867503082197 1.642 0.10874  
## U1 -0.2837258784713196924 0.1556584746932818675 -1.823 0.07622  
## U2 0.4090916262204149501 0.1582795086237918925 2.585 0.01371  
## Ineq 0.6326935328246856560 0.1439940625623334636 4.394 0.0000863344  
## Prob -0.2231607928097278648 0.0876319928046151164 -2.547 0.01505  
##   
## (Intercept)   
## M \*\*   
## Ed \*\*   
## Po1 \*\*\*  
## M.F   
## U1 .   
## U2 \*   
## Ineq \*\*\*  
## Prob \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5056 on 38 degrees of freedom  
## Multiple R-squared: 0.7888, Adjusted R-squared: 0.7444   
## F-statistic: 17.74 on 8 and 38 DF, p-value: 0.0000000001159

This puts our R-squared value around 74% and p-value close to 0

Next we’ll use the caret package – specifically the leaps and the MASS packages to fit our linear regression model using stepwise selection (leapSeq). Set up repeated 10-fold cross-validation, which will let us estimate the RMSE (average prediction error) for each of the 5 models specified by nvmax.

## nvmax RMSE Rsquared MAE RMSESD RsquaredSD MAESD  
## 1 1 285.5037 0.5797686 231.0552 75.93507 0.3766473 64.65062  
## 2 2 251.6307 0.5885311 191.6186 90.14342 0.3475418 60.22092  
## 3 3 247.1283 0.5489734 197.2120 99.55289 0.3557170 77.35470  
## 4 4 261.8166 0.5968752 206.0392 83.13210 0.3131785 69.53834  
## 5 5 246.7830 0.6003050 199.6644 81.81135 0.2985455 60.59434

## We can see that an nvmax of 5 is the best model with an RMSE of 246.783



Our final model and coefficients are:

## Subset selection object  
## 15 Variables (and intercept)  
## Forced in Forced out  
## M FALSE FALSE  
## So FALSE FALSE  
## Ed FALSE FALSE  
## Po1 FALSE FALSE  
## Po2 FALSE FALSE  
## LF FALSE FALSE  
## M.F FALSE FALSE  
## Pop FALSE FALSE  
## NW FALSE FALSE  
## U1 FALSE FALSE  
## U2 FALSE FALSE  
## Wealth FALSE FALSE  
## Ineq FALSE FALSE  
## Prob FALSE FALSE  
## Time FALSE FALSE  
## 1 subsets of each size up to 5  
## Selection Algorithm: 'sequential replacement'  
## M So Ed Po1 Po2 LF M.F Pop NW U1 U2 Wealth Ineq Prob Time  
## 1 ( 1 ) " " " " " " "\*" " " " " " " " " " " " " " " " " " " " " " "   
## 2 ( 1 ) " " " " " " "\*" " " " " " " " " " " " " " " " " "\*" " " " "   
## 3 ( 1 ) " " " " "\*" "\*" " " " " " " " " " " " " " " " " "\*" " " " "   
## 4 ( 1 ) "\*" "\*" "\*" "\*" " " " " " " " " " " " " " " " " " " " " " "   
## 5 ( 1 ) "\*" " " "\*" "\*" " " " " " " " " " " " " " " " " "\*" "\*" " "

## (Intercept) Ed Po1 Ineq   
## -3275.4088 157.8695 124.3143 75.0575

An asterisk indicates that a given variable is going to be included in our final model. For example, since we got the lowest RMSE of **246.783036** value with **5** predictors, we are going to include:

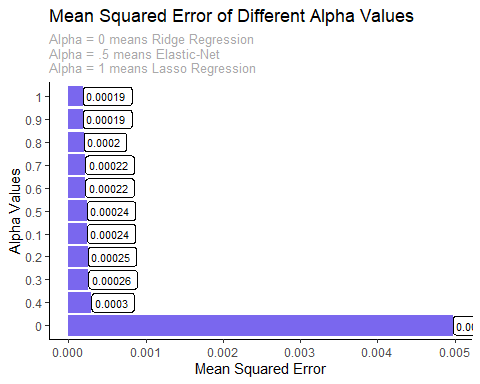
1. Ed (mean years of schooling of the population aged 25 years or over)
2. Po1 (per capita expenditure on police protection in 1960)
3. Ineq (income inequality: percentage of families earning below half the median income)

Thus our final model becomes:

## Crime = -3275.409+ (Ed x 157.8695) + (Po1 x 124.3143) + (Ineq x 75.0575)

## Lasso and Elastic-Net Regression

After doing some research, I found that both variable selection methods use the same glmnet() functions. The only difference is an input parameter, denoted ‘alpha’, which differs between the two. So we’ll just look at them simultaneously, along with a number of other variations of alpha values.



## From above, we can see that an alpha value of 0.9 produces our lowest Mean Squared Error of 0.00019.

### Find our Model

Now that we know which alpha value gives us the lowest Mean Squared Error, we’ll pull that model out and treat that as our linear regression model.

Find the alpha value’s lambda.1se, which is the largest value of lambda such that error is within 1 standard error of the minimum.

## [1] 0.01398437

Run our prediction on our model. Note that s probably refers to the size of the penalty we are setting.

## 1  
## [1,] 0.98059186  
## [2,] 0.01001711  
## [3,] 0.98059186  
## [4,] 0.01001711  
## [5,] 0.01001711  
## [6,] 0.01001711  
## [7,] 0.98059186  
## [8,] 0.98059186  
## [9,] 0.98059186  
## [10,] 0.01001711  
## [11,] 0.01001711  
## [12,] 0.01001711  
## [13,] 0.01001711  
## [14,] 0.01001711  
## [15,] 0.98059186  
## [16,] 0.98059186  
## [17,] 0.01001711  
## [18,] 0.98059186  
## [19,] 0.01001711  
## [20,] 0.01001711  
## [21,] 0.01001711  
## [22,] 0.98059186  
## [23,] 0.01001711  
## [24,] 0.01001711  
## [25,] 0.01001711  
## [26,] 0.01001711  
## [27,] 0.01001711  
## [28,] 0.01001711  
## [29,] 0.01001711  
## [30,] 0.98059186  
## [31,] 0.01001711  
## [32,] 0.01001711  
## [33,] 0.98059186  
## [34,] 0.01001711  
## [35,] 0.01001711  
## [36,] 0.01001711  
## [37,] 0.98059186  
## [38,] 0.01001711  
## [39,] 0.98059186  
## [40,] 0.98059186  
## [41,] 0.01001711  
## [42,] 0.01001711  
## [43,] 0.98059186  
## [44,] 0.01001711  
## [45,] 0.98059186  
## [46,] 0.01001711  
## [47,] 0.01001711

Get our mean squared error

## [1] 0.0001944134

Now that we have our glmnet conducted, we’ll figure out which coefficients we need so we can build out our linear regression model.

## Warning in model.matrix.default(mt, mf, contrasts): the response appeared on the  
## right-hand side and was dropped

## Warning in model.matrix.default(mt, mf, contrasts): problem with term 1 in  
## model.matrix: no columns are assigned

##   
## Call:  
## lm(formula = Crime ~ ., data = new\_crime)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -563.09 -246.59 -74.09 152.41 1087.91   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 905.09 56.42 16.04 <0.0000000000000002 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 386.8 on 46 degrees of freedom

We can see that our Adjusted R-squared value is 66%. Not all of the coefficients seem to have low p-values, and are thus not significant, so I would suggest taking more of a manual approach to sifting through the data and selecting variables.

### Thanks for reading!