HW 6 - Kelly “Scott” Sims

Using the same crime data set uscrime.txt as in Question 8.2, apply Principal Component Analysis and then create a regression model using the first few principal components. Specify your new model in terms of the original variables (not the principal components), and compare its quality to that of your solution to Question 8.2. You can use the R function prcomp for PCA. (Note that to first scale the data, you can include scale. = TRUE to scale as part of the PCA function. Don’t forget that, to make a prediction for the new city, you’ll need to unscale the coefficients (i.e., do the scaling calculation in reverse)!)

Let’s first take a look at our data again

```r
library(ggplot2)
library(ggfortify)
data <- read.table('uscrime.txt', header = TRUE, stringsAsFactors = FALSE)
head(data)
```

```
##      M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  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 25.0
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6
##       Prob    Time Crime
## 1 0.084602 26.2011   791
## 2 0.029599 25.2999  1635
## 3 0.083401 24.3006   578
## 4 0.015801 29.9012  1969
## 5 0.041399 21.2998  1234
## 6 0.034201 20.9995   682
```

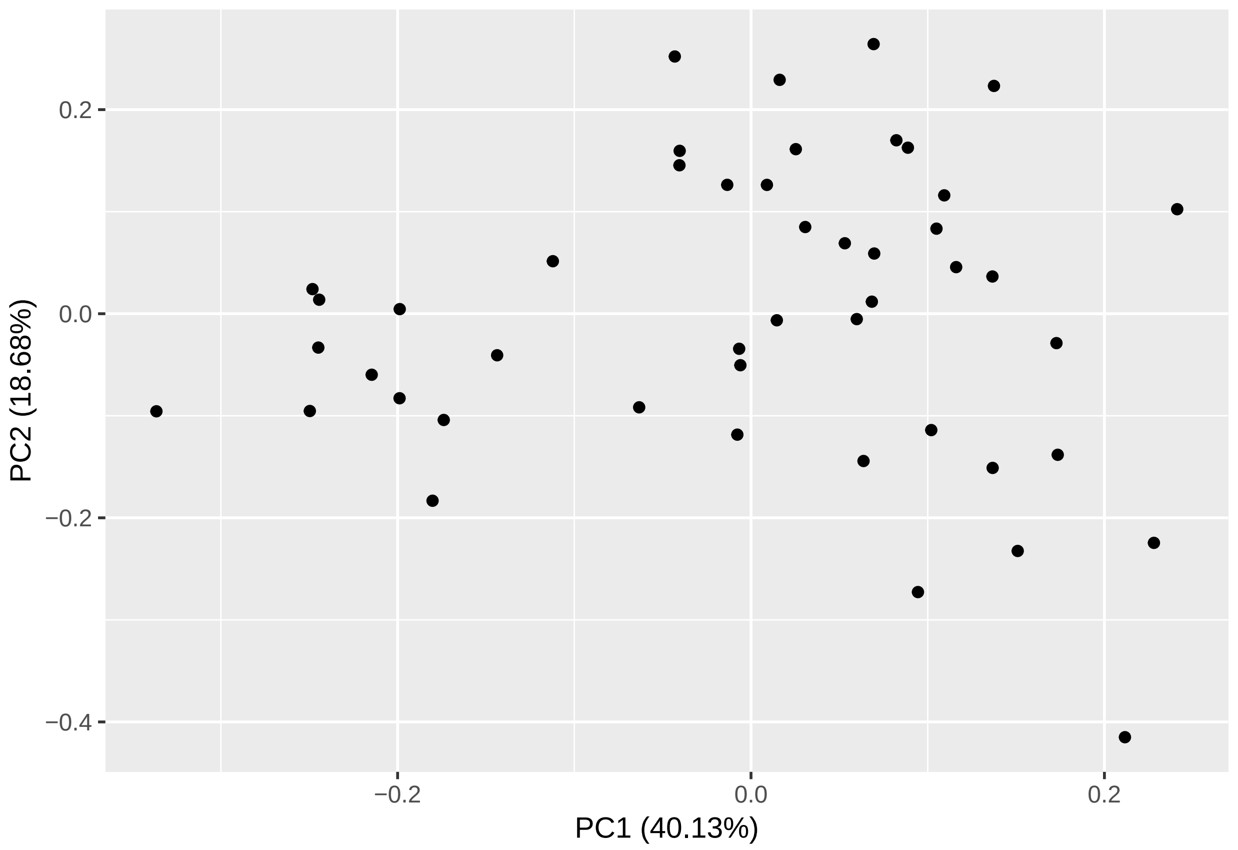
We can see that there are 15 different features in this dataset (not including the target variable, crime), each which attribute to their own dimension in vector space. (15-dimensional graph if you will). Their dimensions between each other vary greatly. Prob is as low as 0.02 where as Wealth is in the thousands. Scaling the data will be necessary. Next, we will investigate the priciple components of this data and see which dimensions explain the most variance

Perform PCA Analysis

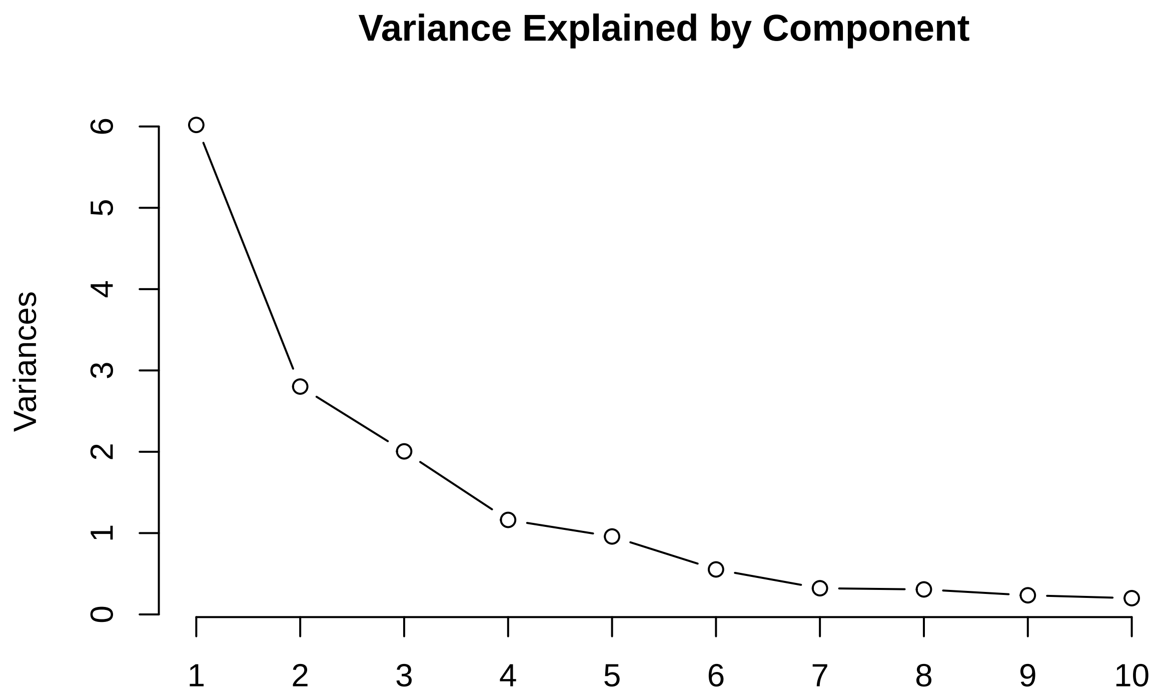
```r
data.pca <- prcomp(data[-16], center = TRUE, scale. = TRUE)
summary(data.pca)
```

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

```
autoplot(data.pca)
```

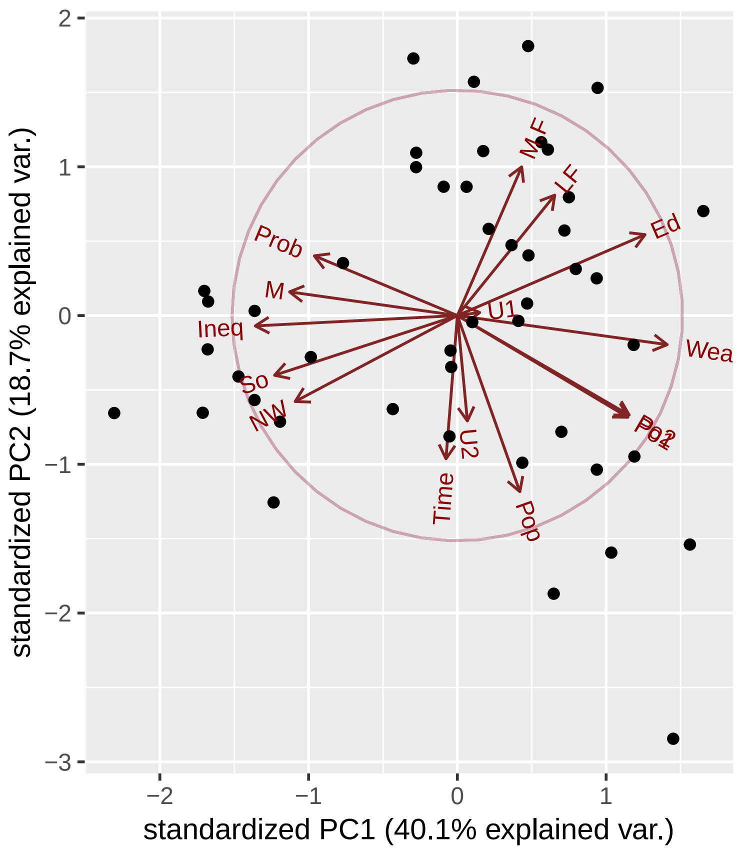


```
plot(data.pca, type = 'line', main = 'Variance Explained by Component')
```



We scaled and center the data while performing PCA, and we can see that we have 15 principle components. Each component explains a certain amount of variance within the data. For instance PC1 and PC2 explains 40% and 19% of the variance respectively. That means from just these two components, nearly 50% of the information in the dataset, of 15 variables, can be explained. The scree plot (variance explained by component) shows all the components (x-axis) and the Eigenvalues (y-axis) which basically stand for the amount of variance each component represent. By using only the first 9 Principle components, we can effectively reduce our model from one that is of 15 features down to 9 while maintaining ~96% of the variance in the data. Next, we will investigate further into the first 2 principle components with a loading plot.

```r
ggbiplot::ggbiplot(data.pca, circle = TRUE)
```



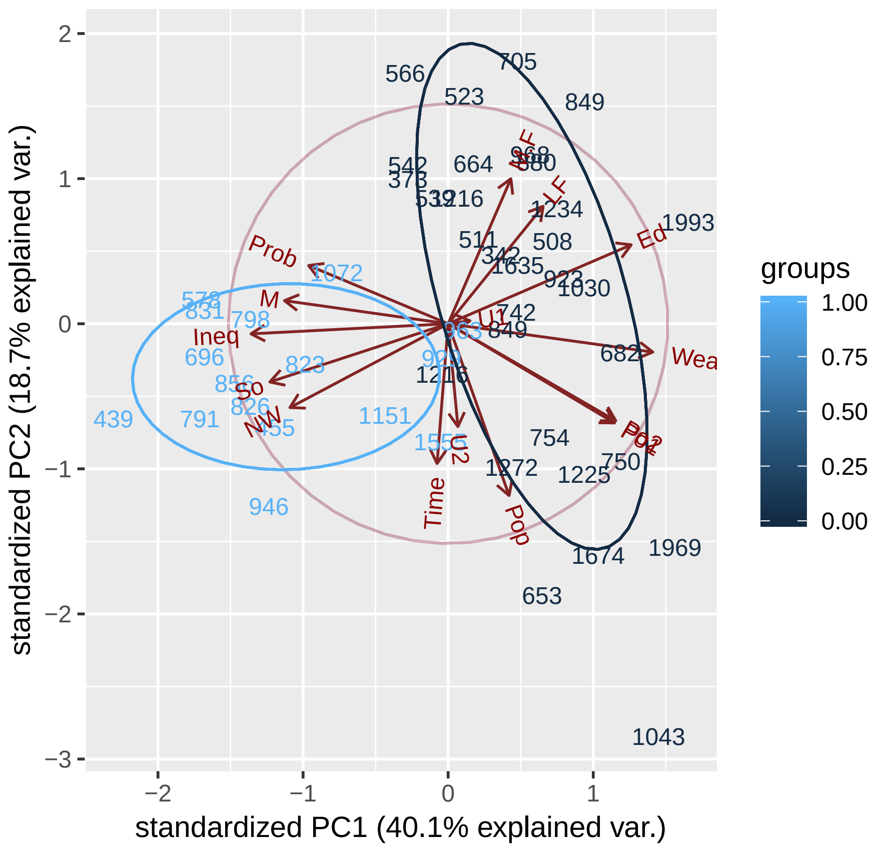
The plot above shows how strongly each feature influences each principle component. We can see that Wealth greatly influences PC1 in the positive direction while Ineq greatly influences PC1 in the negative direction. Since these two features are almost 180 degrees out from each other, they're negatively correlated. Conversely, features PO1 and PO2 are almost on top of each other, meaning they're very positively correlated. Judging from the loading chart, we can see PC1 is in fact mostly a measure of Wealth, ED, PO1 (and PO2) and Ineq, where PC2 is mostly a measure of Pop and M.F.

If we had specific instances of a categorical class we were trying predict, we could use PCA to see how these observations all group together categorically. For instance, if we were trying to determine how crime looked in southern states versus northern states, we could color the data via this binary category and highlight the groupings (NOTE: Since 'So' is a feature being fed into PCA, this is known as data leakage, and in practice should be avoided always. The following graph is merely for illustrative purposes)

```r
library(ggbiplot)
```

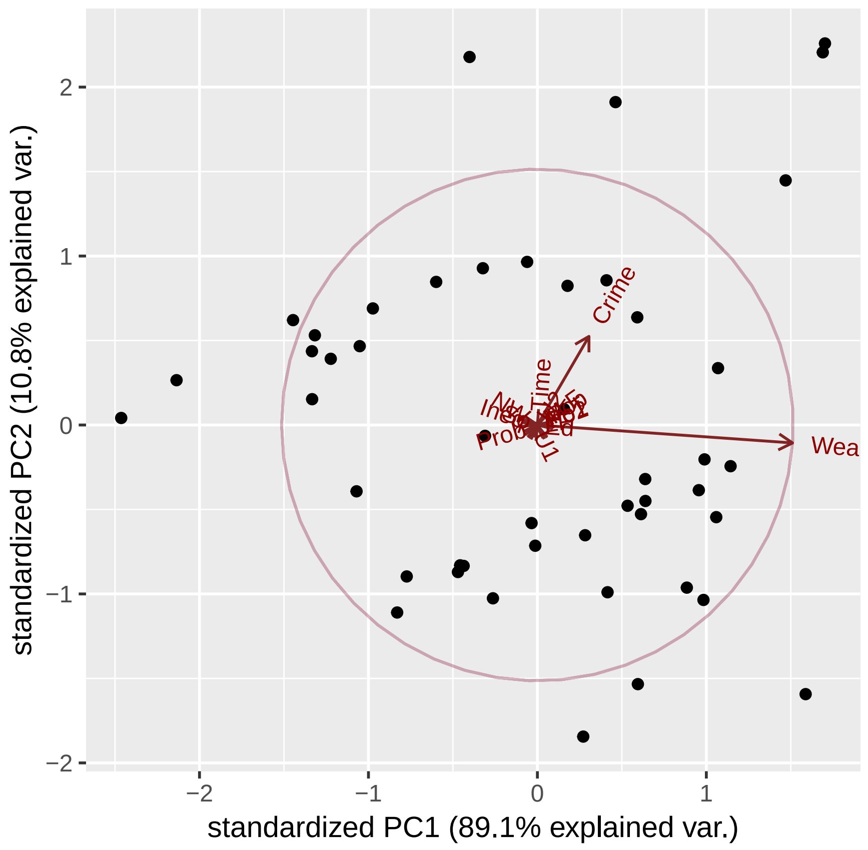
```
##
## Attaching package: 'ggbiplot'

## The following object is masked from 'package:ggfortify':
##
##     ggbiplot
```

```r
ggbiplot::ggbiplot(data.pca, labels = data$Crime, groups=data$So, ellipse = TRUE, circle = TRUE)
```



From the graph above, we can see that Southern states, noted by the light blue color are grouped with each other on the negative side of PC1 while northern states, noted in black, are grouped more so on the positive side. Again this is to be expected since So negatively affects PC1, so being a southern state would group you on the left side of PC1. Hence, data leakage. The red circle in the middle represents the overall magnitude of each feature. Each feature is clearly visible within the circle because we scaled the data. Had we not scaled it, the larger features would greatly dominate the component and it would consume the majority of the variance. Let's visualize this real quick.

```r
unscaled <- prcomp(data[-1])
ggbiplot::ggbiplot(unscaled, circle = TRUE)
```



As you can see, the numerical ranges of Wealth overshadow all other features so much that it dominates the component space. This is why we scale data!! Now that we have fully investigated our principle components, let's build a regression model using the first 5 components. I'm selecting the first 5 even though I previously mentioned the first 9 early, because selecting 9 PCAs could possibly still lead to overfitting. Also, with only 5 PCAs, we are still accounting for ~87% of the variance.

```
#Create a dataframe using the PC values and create a crime column to bring in the target variable
train.data <- data.frame(Crime = data$Crime, data.pca$x)

#We are interested in the first 5 PCAs
train.data <- train.data[,1:6]
head(train.data)
```

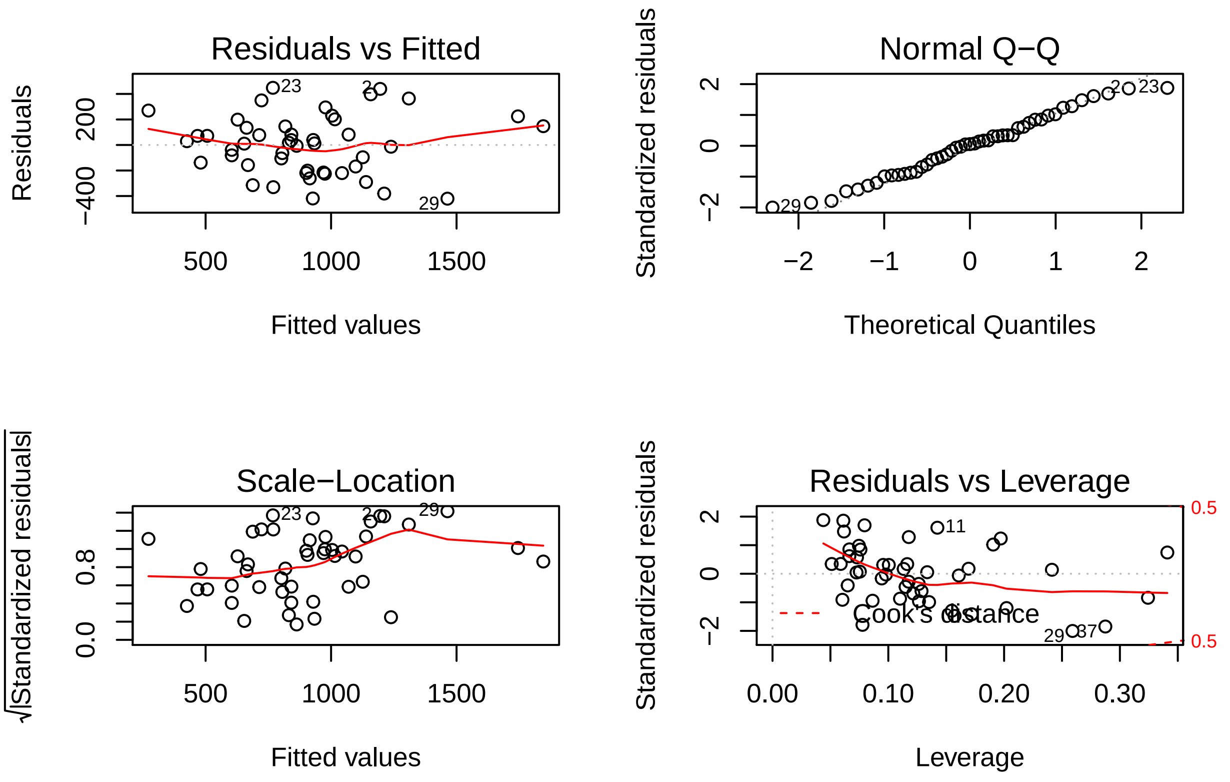
```
##   Crime       PC1        PC2         PC3         PC4         PC5
## 1   791 -4.199284 -1.0938312 -1.11907395  0.67178115  0.05528338
## 2  1635  1.172663  0.6770136 -0.05244634 -0.08350709 -1.17319982
## 3   578 -4.173725  0.2767750 -0.37107658  0.37793995  0.54134525
## 4  1969  3.834962 -2.5769060  0.22793998  0.38262331 -1.64474650
## 5  1234  1.839300  1.3309856  1.27882805  0.71814305  0.04159032
## 6   682  2.907234 -0.3305421  0.53288181  1.22140635  1.37436096
```

Create The Model

```
pca.model <- lm(Crime~., data = train.data)
summary(pca.model)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = train.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -420.79 -185.01   12.21  146.24  447.86
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      35.59  25.428  < 2e-16 ***
## PC1            65.22      14.67   4.447 6.51e-05 ***
## PC2           -70.08      21.49  -3.261  0.00224 **
## PC3            25.19      25.41   0.992  0.32725
## PC4            69.45      33.37   2.081  0.04374 *
## PC5          -229.04      36.75  -6.232 2.02e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 244 on 41 degrees of freedom
## Multiple R-squared:  0.6452, Adjusted R-squared:  0.6019
## F-statistic: 14.91 on 5 and 41 DF,  p-value: 2.446e-08
```

```r
par(mfrow=c(2,2))
plot(pca.model)
```



*__Analysis__ There's a lot going on the in the summary statistics. The main thing to note is the R-squared and the F-statistc. The R-squared is a measure of how much variablitly our model can explain. As seen above, our model can explain ~65% of the variability in the data. This is pretty good. The F-stastic compares our model to an "intercept only model". An intercept only model is one without any additional features. Basically you interpret the F-statistic with a null and alternative hypothesis

> *H0*: The fit of intercept only model and the current model is the same. i.e. Additional variables do not provide value taken together
>
> *H1*: The fit of intercept only model is significantly less compared to our current model. i.e. Additional variables do make the model significantly beter.

In our F-statistic above, we see we have a value of 14.91 with a p-value of 2.446e-08. This p-value is far less than 0.05 (The usual critical value for significance). Therefore, we can conclude that having our 5 PCAs in the model is significant enough that they should be there.

Next, let's predict on unseen data

Unseen data

```
new.data <- data.frame("M"= 14,
"So" = 0,
"Ed" = 10,
"Po1" = 12,
"Po2" = 15.5,
"LF" = .640,
"M.F" = 94,
"Pop" = 150,
"NW" = 1.1,
"U1" = .120,
"U2" = 3.6,
"Wealth" = 3200,
"Ineq" = 20.1,
"Prob" = .04,
"Time" = 39)

#Transform the new data into PCA using the same scaling as the training data
new.datapca <- as.data.frame(predict(data.pca, newdata = new.data))
new.datapca
```

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

> Next we will select the first 5 PCAs of this new data and predict upon it with our regression model

```
new.datapca <- new.datapca[1,1:5]
#This automatically scales the new data with respect to the previously scaled data during the PCA part
predict(pca.model, newdata = new.datapca)
```

```
##        1
## 1388.926
```

This predicted value seems reasonable since the range for crime is 500-2000. Next, we will see what our 5 PCAs are mostly comprised of. I will run last weeks regression model very quickly for comparison purposes without any deep explanation. Once we have the results of the two, we can compare and contrast.

```
sig.model <- lm(Crime ~ Ed + Ineq + M + Prob + U2 + Po1, data = data)
summary(sig.model)
```

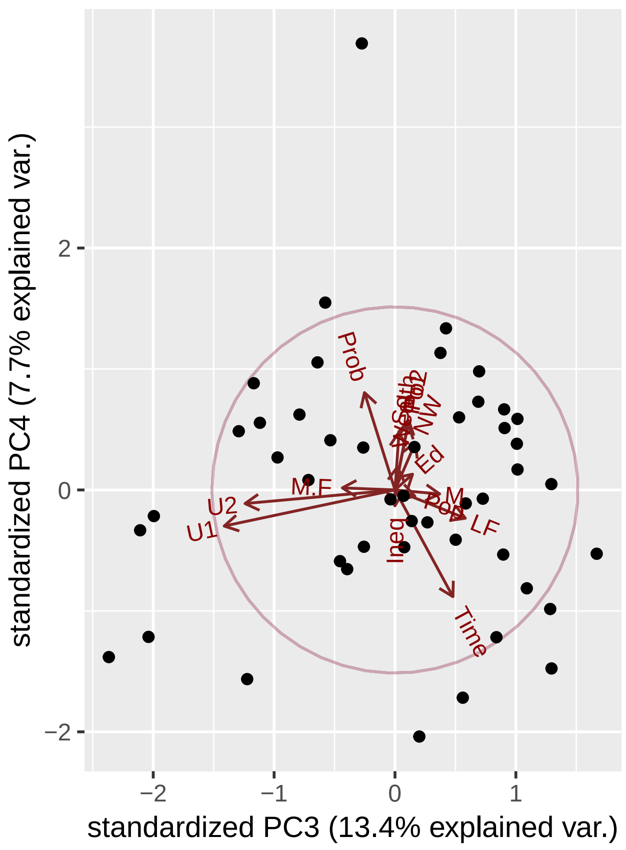
```
##
## Call:
## lm(formula = Crime ~ Ed + Ineq + M + Prob + U2 + Po1, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -470.68  -78.41  -19.68  133.12  556.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5040.50     899.84  -5.602 1.72e-06 ***
## Ed            196.47      44.75   4.390 8.07e-05 ***
## Ineq           67.65      13.94   4.855 1.88e-05 ***
## M             105.02      33.30   3.154  0.00305 **
## Prob        -3801.84    1528.10  -2.488  0.01711 *
## U2             89.37      40.91   2.185  0.03483 *
## Po1           115.02      13.75   8.363 2.56e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200.7 on 40 degrees of freedom
## Multiple R-squared:  0.7659, Adjusted R-squared:  0.7307
## F-statistic: 21.81 on 6 and 40 DF,  p-value: 3.418e-11
```

```
predict.lm(sig.model, new.data)
```

```
##        1
## 1304.245
```

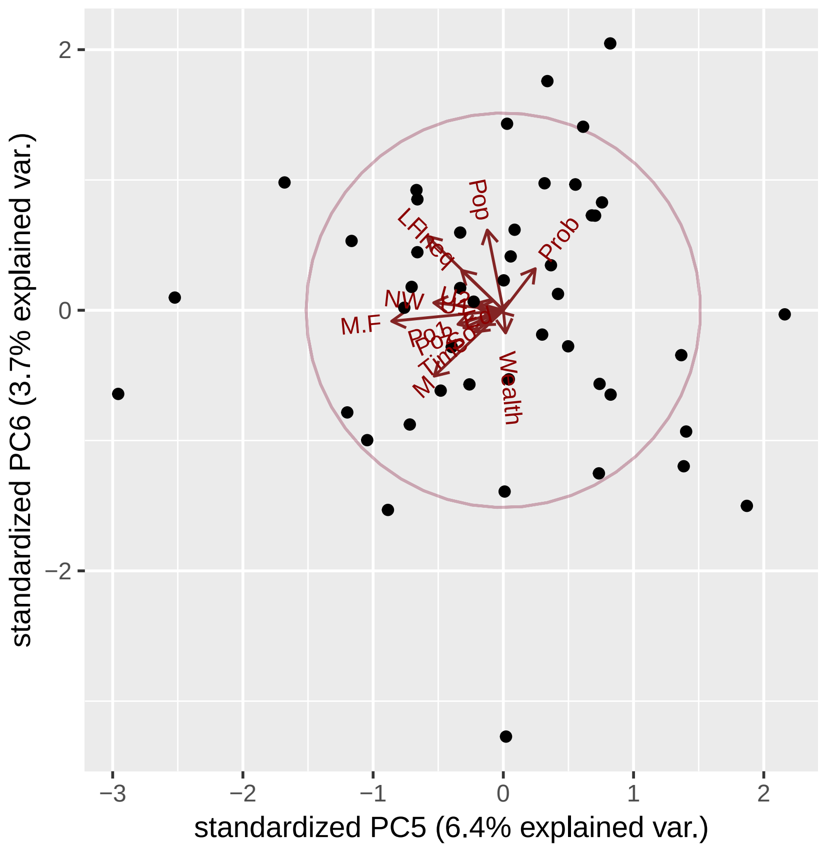
> So when we ran our original regression model last week, we did this really long analysis that ultimately landed us to these 6 main features of Ed, Ineq, M, Prob, U2, and Po1. The R squared value of 0.7659 is really good, but also hints a little towards overfitting. However, the value of 1304.245 seems a reasonable number. With PCA, we didn't have to do a super long analysis to remove highly correlated data and manually select significant features. We were able to reduce our vector space from 15 to 5 while maintaining information and randomness from all of the features. The predicted value from PCA was still in the near realm of the original regression model. We can see what the significant features were for the regression model from last week. We also know that PC1 was characterized mostly by Wealth, ED, PO1 (and PO2) and Ineq, where PC2 is mostly a measure of Pop and M.F. Let's now see what the other 3 PCAs used in the analysis are characterized by.

```
ggbiplot::ggbiplot(data.pca, circle = TRUE, choices = c(3,4))
```



We can see that PC3 is comprised of mostly U1 and U2. Both of these variables are pretty close in proximity and magnitude to each other which hints at a strong colinear relationship. PC4 Is mostly explained by Prob, and Time. Again, these two features are almost 180 degrees out, so they are negatively correlated with respect to PC4

```
ggbiplot::ggbiplot(data.pca, circle = TRUE, choice = c(5,6))
```



Finally PCA5 is mostly characterized by M.F. Almost all of the main characterizing features in PCA were also deemed significant, and included in last weeks regression model. This is a great example of how PCA can be used to subdue overfitting while maintaining information from all features. It is also a way of measuring feature importance by their over all magnitude in characterizing each PCA