PCA: Creating New Ways to Look at Data

Cameron Rider

October 31, 2017

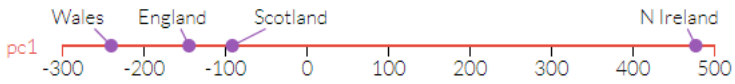
At first glance, the statistical procedure called Principal Component Analysis feels like something that ought to be avoided at all costs. Not only does its Wikipedia page make it seem like some twisted combination of linear algebra and black magic (honestly, is there any difference between the two) - it also takes your perfectly useful sample data and categorizes it under completely new names like…PC1 and PC2. However, Principal Component Analysis (PCA for short) can be a very powerful tool in the hands of the knowledgeable, and it only takes a little reading to join the ranks of those who can use it to their benefit.

Currently, one of the biggest challenges in the field of statistics is figuring out what data is most relevant to the question we want to answer. A company might have a whole slew of information about its customers or users, but without a good way to construct the bigger picture from their millions of data points, it's practically worthless. PCA excels at dealing with those types of data sets: more specifically, it can take a wide range of independent variables and summarize them into fewer, much more impactful categories. By comparing these to our original data, we can easily crossreference which variables seem to have the biggest impact across the board, and which ones we can ignore as negligible. This process decreases the amount of different factors we have to work with as statisticians, and gives us an insight into the data we can't always get by examining the raw numbers. But enough of taking my word for it - let's take a look at some examples:

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

Images can be found at Explained Visually, another wonderful place to learn more about PCA!

This table represents the average consumption of SEVENTEEN different food categories per capita over a week. Even with this nicely organized and labeled barchart, there are so many observations to make that we don't really know where to start. Thankfully, PCA makes it much easier for us to look at variations in the data we've been given. We'll skip the R code for now and look at the first result PCA gives us:



Now, before you accuse me of using black magic like everyone else, let me explain - PCA's power lies in looking at **variation**. By looking through our variables (cheese, fresh fruit, potatoes) and finding the ones with the biggest differences between countries, PCA is able to identify the core differences between each country and summarize it all into just a few variables. Each principal component (PC1, PC2, etc) is simply a linear sum of all the different variables you plug into PCA: it weighs each variable based off its calculated importance, and centers it at zero. PC1 contains the most variation, PC2 covers the variation that PC1 could not cover, PC3 covers what PC2 could not, and so on. Now that that's settled, let's look at the PC1 of each of our countries again.

Immediately, we can see that Northern Ireland varies from the other countries by a much larger margin. A look at our original data confirms this: most notably, they appear to eat much more fresh potatoes than any of the others, while apparently avoiding alcoholic drinks, fresh fruit, and cheese. Mathematically, PC1 is a combination of all of these factors.

Now, how does this help us? Technically, our analysis didn't change a thing. If we had keen enough eyes and had gotten a full 8 hours of sleep the previous night, we might've picked out the major differences between Northern Ireland and its peers without ever touching PCA. However, in cases with even more variables and categories to worry about, PCA becomes an indispensable tool for examining data. Thanks to some quick analysis, we can draw some important conclusions from our data, even without knowing exactly what PC1 stands for. Perhaps it'd be more profitable to stop airing commercials for alcohol in North Ireland and instead market the newest brand of…potatoes? If we want to make predictive models of what people will be eating in the future, perhaps it'd be wiser to do separate analyses for North Ireland and the rest of the UK? PCA makes it much easier to reach these conclusions, and despite the tough theory it's grounded in, it's incredibly easy to use in your own studies.

The Theory Behind PCA

I'll do my best to keep this section light: while the way I explained PCA makes the calculation of each principal component seem trivial, there is a lot of theory behind it that many others on the Internet have explained better than I will. I'll slowly raise the difficulty as I go along, so if you ever get tired of listening to me talk, feel free to skip to the fun part: the application.

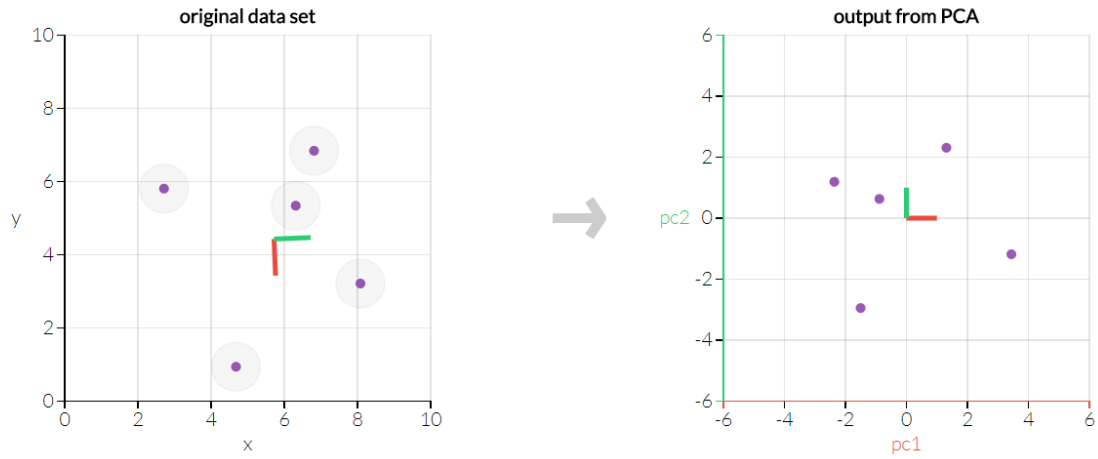
As I mentioned before, PCA takes the variables you're using (cereals, fish, sugars) and converts them into its own variables - one for each of yours. Unlike your system, however, it sorts these variables not by name, but by importance. The way it determines importance is by the amount of variation in each variable. For example, if you're trying to decide which of the three Chinese places near your place is the best, you're not going to base your decision off of each place's white rice. While you might say one place has slightly better white rice than another, chances are there's so little difference between them that they're not even worth comparing. What if you're a kung pao chicken fan, and one place's kung pao chicken is just flat out awful? Even if you don't buy kung pao chicken every time you go out for Chinese, that will be a huge part of your decision. PCA acts the same way - by looking out for large variations in your data samples, it determines what is most "important". A little variation in your data sample can often just mean noise - tiny coincidental differences that don't tell you anything useful. Large variations are meaningful, and allow us to recognize differences or trends in the data.

Thus, after PCA, we have the same amount of variables we started with: however, the first ones are dramatically more important than the ones at the bottom of the ladder. Depending on the number of independent variables you start with, PC1 and PC2 alone could account for the majority of the variance in your sample. We can then see how these principal components were calculated and refer back to our original data to see what mattered most.

Another important aspect of these principal components is that they are completely independent of each other. Now, you may say: "I thought each principal component was a linear sum of all of my original variables! If each principal component takes in all of the original variables to some degree, how could they possibly be independent?". The answer is that the way they are calculated necessitates that they are independent - or, in the more specific linear algebra term, orthogonal to each other.

Principal components are calculated in order of descending importance. PC1 is first calculated to cover as much of the variance in the data as possible in a linear model. PC2 covers as much of the remaining variance as possible, and so on. What "remaining variance" implies is that you can't use any of the previous principal components when building a new one. Thus, while PC3 is built off of the same bases (the original variables in the data) as PC1 and PC2, PC3 cannot be equivalent to a permutation of the linear combinations of PC1 and PC2.

This concept is called "shifting dimensions" in linear algebra. When you're not using PCA, you describe your data in the "dimensions" given to you. In the case of food consumption, these dimensions would be the amount of cheese or fish consumed on average. Through PCA, new dimensions are built, each a combination of all your previous dimensions. Your data hasn't changed - it's simply being described in a different way, one that takes into account cheese and fish and everything else.



Just in case you don't quite see what I mean, here is an example in two dimensions where the difference between the original data set and the PCA output is much more clearly defined. Notice how the PCA output seems to simply be a rotation of the original data, with the red PC1 axis being the line along which the data differs the most? This is what I mean by "shifting dimensions". Of course, this becomes much harder to think about as you add more variables (and thus dimensions), but thankfully we don't have to compute it - the computer does it for you! As long as you have a grasp of what's going on, PCA becomes much easier to work with.

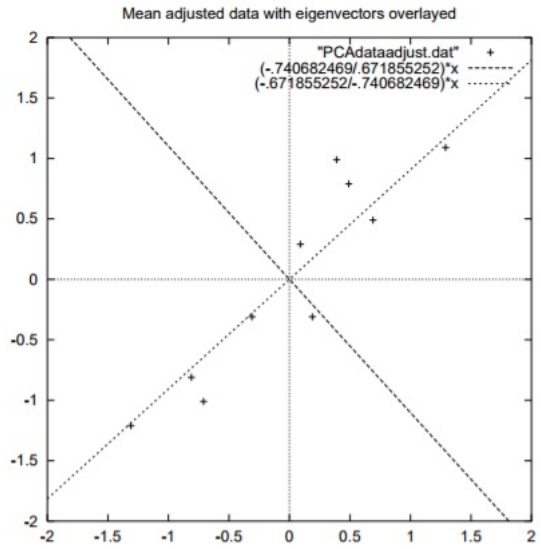
Now we get to the important part, the whole reason we turned to PCA in the first place: "dimensionality reduction", reducing the number of dimensions (read: variables) we have to work with. Technically, this is a misnomer - we never actually just toss out whole variables in our data. Remember, every principal component takes in all of our independent variables to some degree. However, thanks to PCA, we now have a list of the importance of each principal component of our data. If we know that PC1, PC2 and PC3 account for 2/3 of the variation in our data, we can determine how those three components were calculated. This should help us quickly determine what variables in our data we should be looking at.

Finally, let's look into the nitty-gritty of how exactly these principal components are calculated. First off, all of our initial data is normalized, so that the mean of each variable is equal to 0 and the standard deviation is equal to 1. This simply puts all of our data on an equal playing field. Now, our goal for each principal component is to find a linear combination of our variables such that the combined variance, or the covariance, of our data from that linear combination is as high as possible. To keep track of all these different variables, we construct a covariance matrix (in this example, one that takes into account 3 variables):

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

Examples can be found in this tutorial on PCA, which takes you step by step through the theory behind calculating PCA

From there, we can calculate the eigenvectors and eigenvalues of the covariance matrix. (For a brief refresher on how we find eigenvectors and eigenvalues, check out this helpful video I wish I had found when I was taking linear algebra.) The eigenvectors essentially act as lines of best fit for our data, as can be seen in this 2D plot:



Those eigenvectors are our principal components! Now, all we need to do is decide how to order these eigenvectors by importance. Thankfully, that isn't difficult either: it turns out, the higher the eigenvalue of an eigenvector is, the more variation it covers. Thus, the eigenvector with the highest eigenvalue is our PC1, the eigenvector with the next highest eigenvalue is PC2, and so on. We're done!

Applying PCA to your Datasets in R

Let's take a look at a dataset describing the statistics of all the NBA players in the 2017 season based on position.

One common way to determine the "strength" of an individual player is his efficiency, which is a combination of multiple different statistics. The equation for efficiency is shown below:

$$efficiency = (points + rebounds + assists + steals + blocks - missed\_fg - missed\_ft - turnovers)/games\_played$$

where missed_fg is missed 2 or 3 pointers ("field goals"), missed_ft is missed free throws, and the rest are self explanatory.

This problem isn't as simple as it may initially seem. There are a lot of different factors that go into efficiency, and even if a player has the same exact efficiency as another, that doesn't necessarily mean they're doing the same thing on the court. Depending on his position and the team's general strategy, a player could be lacking in some areas but shine in others. Instead of just calculating the efficiency of every player and calling it a day, we can infer much more from the data by utilizing PCA. I have prepared five different data sets, already divided by position, so we can look at how efficiency changes based on a player's position. Here's the one for point guards:

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
library(ggplot2)
```

```
# For more info on what's contained in these data files, feel free to check out the
# source files, data dictionary and R script files in the data and code folders
pointguards <- read.csv("../data/nba2017-pointguard-statistics.csv",
                        stringsAsFactors = FALSE)
# Quick look at our data
head(pointguards, 5)
```

```
##              Player Team Position Experience  Salary Rank Age GP GS  MIN
## 1 Demetrius Jackson  BOS       PG          R 1450000   15  22  5  0   17
## 2     Isaiah Thomas  BOS       PG          5 6587132    1  27 76 76 2569
## 3       Terry Rozier  BOS       PG          1 1906440    9  22 74  0 1263
## 4    Deron Williams  CLE       PG         11  259626   11  32 24  4  486
## 5        Kay Felder  CLE       PG          R  543471   13  21 42  0  386
##   FGM  FGA Points3 Points3_atts Points2 Points2_atts FTM FTA OREB DREB AST
## 1   3    4       1            1       2            3   3   6    2    2   3
## 2 682 1473     245          646     437          827 590 649   43  162 449
## 3 151  411      57          179      94          232  51  66   40  187 131
## 4  68  147      22           53      46           94  21  25    1   44  86
## 5  62  158       7           22      55          136  35  49    3   38  58
##   STL BLK  TO Points  RB Missed_FG Missed_FT       EFF
## 1   0   0   0     10   4         1         3  2.600000
## 2  70  13 210   2199 205       791        59 24.684211
## 3  45  11  47    410 227       260        15  6.783784
## 4   6   6  40    179  45        79         4  8.291667
## 5  18   7  31    166  41        96        14  3.547619
```

As you can see, everything we need for efficiency is already in there, as well as some extra tidbits like players' teams and experience. Let's try doing some PCA analysis with prcomp() and see what we can surmise:

```
# Using dplyr's select function to grab all the columns needed to compute efficiency
PGdata <- select(pointguards, Points3, Points2, FTM, RB, AST, STL,
                 BLK, Missed_FG, Missed_FT, TO)
# prcomp() is the R function dedicated to PCA
pointguardPCA <- prcomp(PGdata, scale. = TRUE)
pointguardPCA
```

```
## Standard deviations (1, .., p=10):
##  [1] 2.8642484 0.8075418 0.6090261 0.5436676 0.4504099 0.3313407 0.2680640
##  [8] 0.2323260 0.1700432 0.1003309
##
## Rotation (n x k) = (10 x 10):
##                 PC1          PC2         PC3         PC4          PC5
## Points3   0.2809429  0.587578621  0.12054575 -0.57521449  0.315101232
## Points2   0.3257612 -0.007297391  0.06373973 -0.06245661 -0.770707033
## FTM       0.3209630  0.242794102  0.37704959  0.27606667  0.111067836
## RB        0.3289418 -0.087032860 -0.09768791  0.21996346  0.247335979
## AST       0.3280676 -0.039912253 -0.44486536  0.19503339 -0.042056569
## STL       0.3109923 -0.137253564 -0.61180075 -0.23442520  0.198940630
## BLK       0.2580429 -0.723169873  0.33922027 -0.46094254  0.113165551
## Missed_FG 0.3374992  0.177481182  0.08567316 -0.17193370 -0.340786116
## Missed_FT 0.3230936 -0.113269912  0.35080428  0.37800365  0.250755814
## TO        0.3383857  0.020942338 -0.10962378  0.25044272  0.006671213
##                    PC6         PC7         PC8         PC9         PC10
## Points3   -0.0539634005  0.13271021 -0.15019380 -0.12819038 -0.267611022
## Points2    0.0075352750 -0.22549595 -0.04203395 -0.04432988 -0.487069058
## FTM        0.3245491611 -0.10207053  0.69625587  0.07482188  0.034603815
## RB        -0.7907192714 -0.30597338  0.14711831 -0.13367871 -0.082159373
## AST        0.1729645737  0.47070533  0.08682052 -0.62660403  0.037630364
## STL        0.3514403184 -0.46153765  0.03400186  0.27679459  0.036247107
## BLK        0.0006859151  0.21728982  0.14881651 -0.02688532 -0.003131921
## Missed_FG -0.1470901560 -0.02992916 -0.12219978  0.04278637  0.813795887
## Missed_FT  0.2801549104 -0.22092156 -0.63593976 -0.14898598 -0.006501001
## TO        -0.1136320702  0.54406250 -0.13808146  0.68127499 -0.134739643
```

There we have it: our ten principal components to match the ten variables we plugged into prcomp(). Now, prcomp() gives us a list of 5 results no matter how many variables we plug into it. These include:

sdev - The standard deviation of the principal components.

rotations - The eigenvectors we were looking for. These are stored in a matrix, where each column is a principal component and each row relates back to our original variables.

center and scale - The centering and scaling used for each original variable.

x - The value of the rotated data in terms of the principal components. Each entry in the first vector corresponds to a player from the initial data set. Within that entry is the player's corresponding value in terms of the principal components.

We can also call summary(pointguardPCA) to see how much variation each principal component covers:

```
# summary() displays the standard deviation and variance proportions of PCA lists
summary(pointguardPCA)
```

```
## Importance of components%s:
##                           PC1     PC2     PC3     PC4     PC5     PC6
## Standard deviation     2.8642 0.80754 0.60903 0.54367 0.45041 0.33134
## Proportion of Variance 0.8204 0.06521 0.03709 0.02956 0.02029 0.01098
## Cumulative Proportion  0.8204 0.88560 0.92270 0.95225 0.97254 0.98352
##                           PC7    PC8     PC9    PC10
## Standard deviation     0.26806 0.2323 0.17004 0.10033
## Proportion of Variance 0.00719 0.0054 0.00289 0.00101
## Cumulative Proportion  0.99070 0.9961 0.99899 1.00000
```

This is an interesting case: in this example, 82% of the variation is already covered by only one principal component, PC1. Furthermore, when we look at what PC1 is composed of, it appears to be a relatively equal spread of all these different stats. That's okay! If we're looking at 85 different point guards, there's going to be a lot of variation across the board. We can look at other parts of the data to see where else we can draw conclusions. For example, let's look at PC7 and PC9:

```
pointguardPCA$rotation[ , c(7, 9)]
```

```
##                    PC7         PC9
## Points3    0.13271021 -0.12819038
## Points2   -0.22549595 -0.04432988
## FTM       -0.10207053  0.07482188
## RB        -0.30597338 -0.13367871
## AST        0.47070533 -0.62660403
## STL       -0.46153765  0.27679459
## BLK        0.21728982 -0.02688532
## Missed_FG -0.02992916  0.04278637
## Missed_FT -0.22092156 -0.14898598
## TO         0.54406250  0.68127499
```

We quickly notice that turnovers (TO) are a major component of both PC7 and PC9. The farther away from 0 and the closer to 1 or -1 a piece of a

component is, the more of the principal component it makes up. Since we know PC7 and PC9 combined make up for less than 1% of the variation in our sample, we can conclude that turnovers won't make up a major part of a point guard's effectiveness. You can make the same conclusion for missed field goals in PC10 or free throws in PC8. While we're not deciding to entirely throw out these statistics, we can now keep in mind that those variables don't have as big an effect on a point guard's effectiveness.

Let's try a different position this time. How about centers?

```
# The same process we went for the point guard data
centers <- read.csv("../data/nba2017-center-statistics.csv",
                    stringsAsFactors = FALSE)
head(centers, 5)
```

```
##          Player Team Position Experience   Salary Rank Age GP GS  MIN FGM
## 1    Al Horford  BOS        C          9 26540100    4  30 68 68 2193 379
## 2  Kelly Olynyk  BOS        C          3  3094014    7  25 75  6 1538 260
## 3  Tyler Zeller  BOS        C          4  8000000   12  27 51  5  525  78
## 4 Channing Frye  CLE        C         10  7806971    7  33 74 15 1398 238
## 5   Edy Tavares  CLE        C          1     5145   18  24  1  0   24   3
##   FGA Points3 Points3_atts Points2 Points2_atts FTM FTA OREB DREB AST STL
## 1 801      86          242     293          559 108 135   95  369 337  52
## 2 508      68          192     192          316  90 123   72  288 148  43
## 3 158       0            1      78          157  22  39   43   81  42   7
## 4 520     137          335     101          185  63  74   37  253  45  33
## 5   4       0            0       3            4   0   1    4    6   1   0
##   BLK  TO Points  RB Missed_FG Missed_FT       EFF
## 1  87 116    952 464       422        27 19.514706
## 2  29  96    678 360       248        33 11.746667
## 3  21  20    178 124        80        17  5.000000
## 4  37  53    676 290       282        11  9.932432
## 5   6   2      6  10         1         1 19.000000
```

```
Cdata <- select(centers, Points3, Points2, FTM, RB, AST, STL,
                BLK, Missed_FG, Missed_FT, TO)
centerPCA <- prcomp(Cdata, scale. = TRUE)
centerPCA
```

```
## Standard deviations (1, .., p=10):
##  [1] 2.6635129 1.1784364 0.6855510 0.5489629 0.4857858 0.4590633 0.3846084
##  [8] 0.2773284 0.2349505 0.1374146
##
## Rotation (n x k) = (10 x 10):
##                  PC1         PC2         PC3         PC4          PC5
## Points3   -0.1615498  0.68860637 -0.38719423  0.5196561 -0.051535585
## Points2   -0.3605091 -0.06315262  0.02658898 -0.3025302  0.248839136
## FTM       -0.3441952  0.00957518 -0.31638767 -0.2553885  0.303101704
## RB        -0.3386688 -0.26391017  0.04896048  0.1160782 -0.104563337
## AST       -0.2830403  0.37589310  0.53511615 -0.1701105 -0.561592330
## STL       -0.3204235 -0.08094628  0.49334663  0.3905143  0.294150414
## BLK       -0.3161112 -0.19029323 -0.43676642 -0.2027607 -0.538412287
## Missed_FG -0.3393402  0.27581102 -0.06102218 -0.1643932  0.352964488
## Missed_FT -0.2861983 -0.43744287 -0.12504293  0.5460287 -0.127374928
## TO        -0.3619355  0.02460360  0.08373072 -0.1096003  0.003539709
##                   PC6         PC7         PC8          PC9         PC10
## Points3    0.008678618 -0.09995508 -0.13438323 -0.046661571  0.216339369
## Points2    0.111299604 -0.29885398  0.21028063 -0.005305799  0.752293249
## FTM        0.131579032  0.60224017 -0.16162918 -0.457428952 -0.094188041
## RB         0.193099142 -0.57932790 -0.41051280 -0.428913298 -0.251901989
## AST        0.189750691  0.18318056  0.19806949 -0.197013799 -0.017700587
## STL       -0.602666324  0.16984987 -0.09561357 -0.054263433  0.064942369
## BLK       -0.574981201 -0.00940716  0.04486473  0.095900775  0.013997161
## Missed_FG -0.072490198 -0.28376339  0.47745426  0.179976819 -0.554358224
## Missed_FT  0.376307637  0.20688442  0.44247311  0.128529105 -0.003998686
## TO         0.237189984  0.12870825 -0.51860111  0.710436050 -0.053575862
```

Now we start to see even more differences in the data! In PC1, we see much less of a focus on 3 pointers, and greater variation in areas such as 2 pointers, rebounds, free throws, and turnovers. Let's check where the variation lies in this sample:

```
summary(centerPCA)
```

```
## Importance of components%s:
##                           PC1    PC2    PC3     PC4    PC5     PC6     PC7
## Standard deviation     2.6635 1.1784 0.6856 0.54896 0.4858 0.45906 0.38461
## Proportion of Variance 0.7094 0.1389 0.0470 0.03014 0.0236 0.02107 0.01479
## Cumulative Proportion  0.7094 0.8483 0.8953 0.92544 0.9490 0.97011 0.98490
##                           PC8     PC9     PC10
## Standard deviation     0.27733 0.23495 0.13741
## Proportion of Variance 0.00769 0.00552 0.00189
## Cumulative Proportion  0.99259 0.99811 1.00000
```

This time, a lot less of the variation can be covered in the first principal component. This allows us to take PC2 and PC3 more seriously, as well as make better conclusions from them since they won't be as uniform as the PCA analysis for point guards. Let's take a look at those again:

```
# The first three columns of rotation
centerPCA$rotation[, 1:3]
```

```
##                  PC1         PC2         PC3
## Points3   -0.1615498  0.68860637 -0.38719423
## Points2   -0.3605091 -0.06315262  0.02658898
## FTM       -0.3441952  0.00957518 -0.31638767
## RB        -0.3386688 -0.26391017  0.04896048
## AST       -0.2830403  0.37589310  0.53511615
## STL       -0.3204235 -0.08094628  0.49334663
## BLK       -0.3161112 -0.19029323 -0.43676642
## Missed_FG -0.3393402  0.27581102 -0.06102218
## Missed_FT -0.2861983 -0.43744287 -0.12504293
## TO        -0.3619355  0.02460360  0.08373072
```

While we still can't draw any major conclusions from PC1 (the differences between the highest and lowest values are only .2, after all), we can start to get a better sense of how we should look at the data for centers. 2 pointers seem to be more important, as do free throws, rebounds, steals, blocks, missed field goals and turnovers. Now that's 7/10 of our categories: we haven't exactly narrowed down a lot. Perhaps if we looked at the other end of our data…

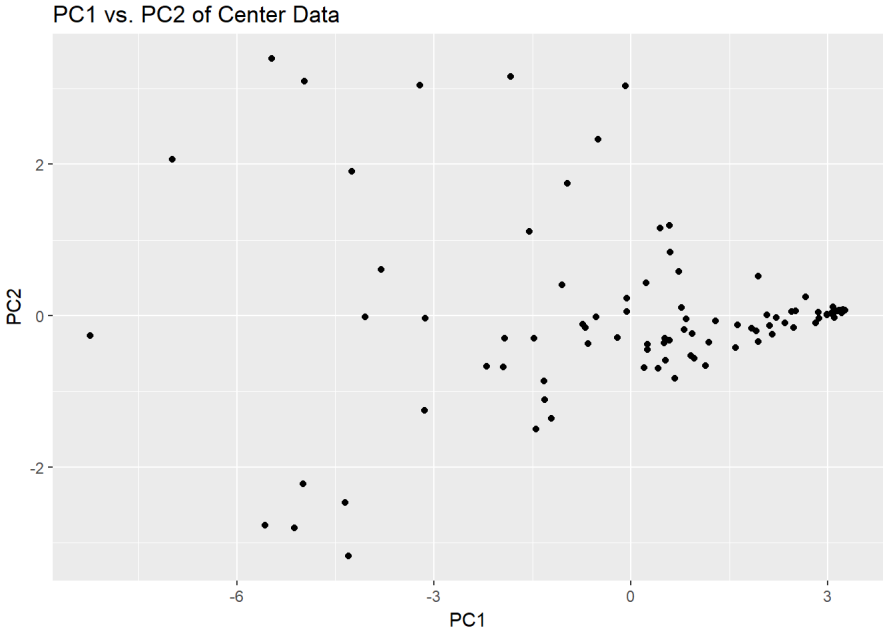
```
centerPCA$rotation[, 8:10]
```

```
##                   PC8          PC9         PC10
## Points3   -0.13438323 -0.046661571  0.216339369
## Points2    0.21028063 -0.005305799  0.752293249
## FTM       -0.16162918 -0.457428952 -0.094188041
## RB        -0.41051280 -0.428913298 -0.251901989
## AST        0.19806949 -0.197013799 -0.017700587
## STL       -0.09561357 -0.054263433  0.064942369
## BLK        0.04486473  0.095900775  0.013997161
## Missed_FG  0.47745426  0.179976819 -0.554358224
## Missed_FT  0.44247311  0.128529105 -0.003998686
## TO        -0.51860111  0.710436050 -0.053575862
```

Huh - a large portion of the variation in 2 points score seems to be quantified in PC10, our least important component. Additionally, turnovers are a major portion of PC9. As we saw in our summary earlier, both PC9 and PC10 account for less than 1% of variation in our data. Perhaps, despite initial impressions, scoring 2 pointers or preventing turnovers isn't as important as a center? These are the observations PCA helps us make, in order to get a good sense of the big picture of our data.

Finally, let's plot all of our centers based on PC1 and PC2:

```
# Graphing PC1 vs. PC2 using ggplot - if you don't know how to use ggplot, plot() works just fine too
ggplot() + geom_point(aes(x = centerPCA$x[, 1], y = centerPCA$x[, 2])) + ggtitle("PC1 vs. PC2 of Center Data") + x
lab("PC1") + ylab("PC2")
```



This sheds even more light on our PCA analysis: a lot of the variation in PC1 appears to be due to fringe points, and not necessarily because of differing playstyles. The heavy appearance of 3 pointers in PC2 also seems to be due to these data points. There's a lot of reasons why our data might look this way. Perhaps those points are less experienced players, or players who rarely get on the court and happened to score a few more/less threes than usual when they had the chance. Maybe these players are all from a small selection of teams: in that case, perhaps those teams like to let their centers move around more and take farther shots? PCA can give us these clues when we're taking a deeper look into our data.

Thanks for bearing with me and I hope you found this post useful! PCA can definitely be difficult to approach at first but using it for your own analysis is extremely easy when you get the hang of what's going on. If you want another explanation or more resources for using PCA, please check my references below:

Principal Component Analysis Explained Visually, which provided the images I used in my intro as well as the 2D comparison of a small data set. vs its principal components. A super simple and very visually pleasing manual for learning how PCA works, and it comes with interactive 2D and 3D graphs!

A One-Stop Shop for Principal Component Analysis, a light but in-depth look into the world of PCA. Includes an indepth explanation on how the principle components are calculated as well as when you should use PCA.

Explaining PCA to Your Family, Great-grandmother to Daughter, a stellar explanation of PCA that begins with how it works at a very basic level and works its way through the theory behind it.

Plots in R using PCA, an article that shows all the different ways you can visualize data after running it through PCA. I didn't get to go through all of these so it's definitely worth a read!

A Mathematical Tutorial on PCA, a guide that goes through the entire process of using linear algebra to calculate the eigenvectors and eigenvalues needed for PCA.

Eigenvectors and eigenvalues, a YouTube visual guide on how eigenvectors act as the line of best fit as we change dimensions.

ng and Visualizing PCA in R, a straightforward guide to using PCA in R and plotting your data in new ways.