project5

##Introduction

Open Pyschometrics is an online repository of interactive personality tests that aggregates, and publishes, anonymous results of those who take the personality accessments. They contain more popular quizzes like the Big Five Personality Test, the MBTI test, and the Ennegram and also lesser known personality tests, such as the Dark Triad test, the Rosenberg Self Esteem Scale, and Exposure Based Face Memory test.

This project explores the correlations of how people answer a personality test to their resulting Emotional quotient and Systematic quotient results. Emotional Quotient, or EQ, is a measure of sensitivity, self-regulation, and societal interaction. Systematic Quotient, or SQ, is a measure of systematic thinking, or logical thinking, skills, distinct and more specific than the IQ.

The test has 120 questions: 60 EQ questions, and 60 SQ questions. We will utilize Principal Components Analysis to see what groups of questions best predict good EQ and SQ scores respectively.

First, our data:

#import our data  
data = na.omit(read\_delim("C:\\Users\\saipr\\Desktop\\Programming\\R\\EQSQ\\data.csv", delim = "\t"))

## Rows: 13256 Columns: 125

## -- Column specification --------------------------------------------------------  
## Delimiter: "\t"  
## dbl (125): E1, E2, E3, E4, E5, E6, E7, E8, E9, E10, E11, E12, E13, E14, E15,...

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

head(data) #view it

## # A tibble: 6 x 125  
## E1 E2 E3 E4 E5 E6 E7 E8 E9 E10 E11 E12 E13  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 3 1 3 2 4 3 1 3 1 3 1 1 4  
## 2 4 3 3 2 2 2 2 3 4 4 3 4 2  
## 3 3 3 3 1 1 2 2 2 3 1 4 2 2  
## 4 2 2 2 1 2 1 3 2 4 4 2 3 4  
## 5 3 1 2 4 4 2 3 3 2 3 1 1 4  
## 6 2 1 3 2 2 3 1 3 4 4 3 1 2  
## # ... with 112 more variables: E14 <dbl>, E15 <dbl>, E16 <dbl>, E17 <dbl>,  
## # E18 <dbl>, E19 <dbl>, E20 <dbl>, E21 <dbl>, E22 <dbl>, E23 <dbl>,  
## # E24 <dbl>, E25 <dbl>, E26 <dbl>, E27 <dbl>, E28 <dbl>, E29 <dbl>,  
## # E30 <dbl>, E31 <dbl>, E32 <dbl>, E33 <dbl>, E34 <dbl>, E35 <dbl>,  
## # E36 <dbl>, E37 <dbl>, E38 <dbl>, E39 <dbl>, E40 <dbl>, E41 <dbl>,  
## # E42 <dbl>, E43 <dbl>, E44 <dbl>, E45 <dbl>, E46 <dbl>, E47 <dbl>,  
## # E48 <dbl>, E49 <dbl>, E50 <dbl>, E51 <dbl>, E52 <dbl>, E53 <dbl>, ...

E1 - E60 are the 60 EQ questions, and S1 - S60 are the 60 SQ questions. Factors like gender, age, and accuracy (one’s self acessment of how accurate they think their scores are) are omitted for the sake of this analysis. We’ll also need to scale our data.

#gets rid of demographic information, results  
X = data %>% select(starts\_with(c("E", "S"))) %>% select(-c("EQ", "SQ"))   
  
#scale our data  
X = as.data.frame(scale(X))  
head(X)

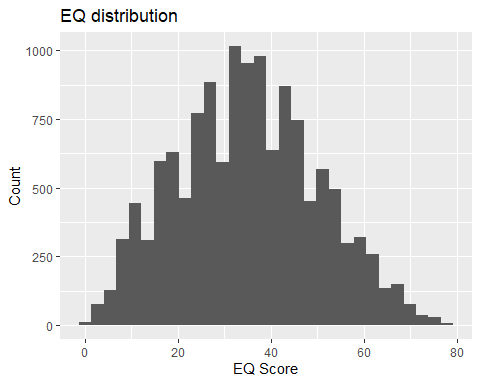
## E1 E2 E3 E4 E5 E6 E7  
## 1 0.4326813 -0.9382649 0.2540536 -0.2563288 1.3062407 0.4867383 -0.99970629  
## 2 1.4629713 1.5154024 0.2540536 -0.2563288 -0.6199539 -0.4933928 0.01702737  
## 3 0.4326813 1.5154024 0.2540536 -1.2791747 -1.5830513 -0.4933928 0.01702737  
## 4 -0.5976086 0.2885687 -0.9478502 -1.2791747 -0.6199539 -1.4735240 1.03376103  
## 5 0.4326813 -0.9382649 -0.9478502 1.7893630 1.3062407 -0.4933928 1.03376103  
## 6 -0.5976086 -0.9382649 0.2540536 -0.2563288 -0.6199539 0.4867383 -0.99970629  
## E8 E9 E10 E11 E12 E13 E14  
## 1 0.3926852 -1.4520936 0.160318 -1.2463657 -1.12474204 1.2519548 1.8945960  
## 2 0.3926852 1.7321831 1.207718 0.7980349 2.10243988 -0.8336139 -1.1925258  
## 3 -0.6481936 0.6707575 -1.934482 1.8202352 -0.04901473 -0.8336139 -0.1634852  
## 4 -0.6481936 1.7321831 1.207718 -0.2241654 1.02671257 1.2519548 0.8655554  
## 5 0.3926852 -0.3906680 0.160318 -1.2463657 -1.12474204 1.2519548 -1.1925258  
## 6 0.3926852 1.7321831 1.207718 0.7980349 -1.12474204 -0.8336139 -0.1634852  
## E15 E16 E17 E18 E19 E20  
## 1 1.4298008 -0.008413867 0.6411669 -1.09870045 -0.4247681 0.9397979  
## 2 -1.3935925 -0.008413867 -0.4541048 2.45560846 0.6127713 -2.0472451  
## 3 0.4886697 -0.008413867 0.6411669 0.08606919 -1.4623075 -2.0472451  
## 4 -0.4524614 -0.008413867 -0.4541048 0.08606919 -0.4247681 0.9397979  
## 5 1.4298008 -0.008413867 -1.5493764 -1.09870045 0.6127713 0.9397979  
## 6 -0.4524614 -0.008413867 -0.4541048 -1.09870045 1.6503107 0.9397979  
## E21 E22 E23 E24 E25 E26 E27  
## 1 0.6922748 1.2408472 -1.3134978 -1.01118715 0.4466501 -0.4054461 0.5839095  
## 2 0.6922748 1.2408472 0.7401351 2.29361978 -1.5529427 1.7599823 1.6284850  
## 3 -1.2247493 0.1205182 1.7669516 0.09041516 -1.5529427 1.7599823 -0.4606660  
## 4 -0.2662373 -2.1201398 -1.3134978 -1.01118715 0.4466501 -1.4881603 -0.4606660  
## 5 -0.2662373 1.2408472 -1.3134978 0.09041516 0.4466501 -1.4881603 0.5839095  
## 6 -0.2662373 0.1205182 -1.3134978 0.09041516 -0.5531463 0.6772681 -0.4606660  
## E28 E29 E30 E31 E32 E33 E34  
## 1 0.1662918 2.037201 0.3483267 -0.3073885 -0.98561979 1.3086619 1.6410617  
## 2 0.1662918 1.024293 -1.7006720 1.7763453 0.09819406 -1.5893706 -1.4883974  
## 3 1.2174906 -1.001523 0.3483267 -0.3073885 0.09819406 -1.5893706 -1.4883974  
## 4 0.1662918 -1.001523 -1.7006720 -0.3073885 2.26582174 -0.6233597 0.5979086  
## 5 -1.9361058 1.024293 1.3728260 -0.3073885 -0.98561979 1.3086619 1.6410617  
## 6 -1.9361058 1.024293 -0.6761727 0.7344784 -0.98561979 0.3426511 0.5979086  
## E35 E36 E37 E38 E39 E40 E41  
## 1 -0.2650322 -0.4452195 1.7382453 1.6053695 1.2589067 -1.4408345 -0.315485  
## 2 -0.2650322 1.5673397 -1.3723401 -0.3976215 -0.9776499 1.8281697 -0.315485  
## 3 -0.2650322 -0.4452195 -1.3723401 1.6053695 -0.9776499 1.8281697 -1.366256  
## 4 -1.4365117 0.5610601 -0.3354783 -1.3991170 -0.9776499 -0.3511664 -0.315485  
## 5 -0.2650322 0.5610601 0.7013835 0.6038740 1.2589067 -0.3511664 0.735286  
## 6 0.9064473 1.5673397 -0.3354783 1.6053695 0.1406284 -0.3511664 -0.315485  
## E42 E43 E44 E45 E46 E47 E48  
## 1 1.4507537 -1.1657036 -0.2995858 -0.7231952 1.07141735 -0.3595538 1.03215321  
## 2 1.4507537 -0.1199154 -0.2995858 1.3531911 0.05563217 0.7566629 -0.06382748  
## 3 0.3598699 1.9716612 0.8363723 1.3531911 0.05563217 1.8728797 -1.15980817  
## 4 0.3598699 -0.1199154 -1.4355440 0.3149980 0.05563217 0.7566629 1.03215321  
## 5 -0.7310139 -1.1657036 0.8363723 0.3149980 1.07141735 -1.4757706 1.03215321  
## 6 -0.7310139 -0.1199154 -0.2995858 -1.7613884 0.05563217 -0.3595538 -0.06382748  
## E49 E50 E51 E52 E53 E54 E55  
## 1 1.5360945 -0.7815061 -1.5740365 1.6854243 -0.5550375 1.6289016 0.4235176  
## 2 -1.3132644 0.2835340 0.4057934 0.6531282 1.5769755 -0.3525363 0.4235176  
## 3 -1.3132644 0.2835340 1.3957084 0.6531282 -1.6210440 0.6381827 -1.7820015  
## 4 1.5360945 -0.7815061 0.4057934 -0.3791679 0.5109690 -0.3525363 0.4235176  
## 5 0.5863082 1.3485741 -0.5841215 -1.4114640 1.5769755 -1.3432553 0.4235176  
## 6 -1.3132644 0.2835340 0.4057934 -0.3791679 0.5109690 -0.3525363 0.4235176  
## E56 E57 E58 E59 E60 S1 S2  
## 1 -0.4226726 -1.4242136 0.9385944 1.069148 0.4120159 0.4326813 -0.9382649  
## 2 1.7343813 1.8729180 -0.1728946 1.069148 -0.6330823 1.4629713 1.5154024  
## 3 0.6558544 1.8729180 -0.1728946 -1.385624 1.4571141 0.4326813 1.5154024  
## 4 -0.4226726 0.7738741 -0.1728946 1.069148 -1.6781805 -0.5976086 0.2885687  
## 5 -0.4226726 -1.4242136 -0.1728946 1.069148 -1.6781805 0.4326813 -0.9382649  
## 6 0.6558544 -0.3251697 0.9385944 1.069148 -0.6330823 -0.5976086 -0.9382649  
## S3 S4 S5 S6 S7 S8 S9  
## 1 0.2540536 -0.2563288 1.3062407 0.4867383 -0.99970629 0.3926852 -1.4520936  
## 2 0.2540536 -0.2563288 -0.6199539 -0.4933928 0.01702737 0.3926852 1.7321831  
## 3 0.2540536 -1.2791747 -1.5830513 -0.4933928 0.01702737 -0.6481936 0.6707575  
## 4 -0.9478502 -1.2791747 -0.6199539 -1.4735240 1.03376103 -0.6481936 1.7321831  
## 5 -0.9478502 1.7893630 1.3062407 -0.4933928 1.03376103 0.3926852 -0.3906680  
## 6 0.2540536 -0.2563288 -0.6199539 0.4867383 -0.99970629 0.3926852 1.7321831  
## S10 S11 S12 S13 S14 S15  
## 1 0.160318 -1.2463657 -1.12474204 1.2519548 1.8945960 1.4298008  
## 2 1.207718 0.7980349 2.10243988 -0.8336139 -1.1925258 -1.3935925  
## 3 -1.934482 1.8202352 -0.04901473 -0.8336139 -0.1634852 0.4886697  
## 4 1.207718 -0.2241654 1.02671257 1.2519548 0.8655554 -0.4524614  
## 5 0.160318 -1.2463657 -1.12474204 1.2519548 -1.1925258 1.4298008  
## 6 1.207718 0.7980349 -1.12474204 -0.8336139 -0.1634852 -0.4524614  
## S16 S17 S18 S19 S20 S21  
## 1 -0.008413867 0.6411669 -1.09870045 -0.4247681 0.9397979 0.6922748  
## 2 -0.008413867 -0.4541048 2.45560846 0.6127713 -2.0472451 0.6922748  
## 3 -0.008413867 0.6411669 0.08606919 -1.4623075 -2.0472451 -1.2247493  
## 4 -0.008413867 -0.4541048 0.08606919 -0.4247681 0.9397979 -0.2662373  
## 5 -0.008413867 -1.5493764 -1.09870045 0.6127713 0.9397979 -0.2662373  
## 6 -0.008413867 -0.4541048 -1.09870045 1.6503107 0.9397979 -0.2662373  
## S22 S23 S24 S25 S26 S27 S28  
## 1 1.2408472 -1.3134978 -1.01118715 0.4466501 -0.4054461 0.5839095 0.1662918  
## 2 1.2408472 0.7401351 2.29361978 -1.5529427 1.7599823 1.6284850 0.1662918  
## 3 0.1205182 1.7669516 0.09041516 -1.5529427 1.7599823 -0.4606660 1.2174906  
## 4 -2.1201398 -1.3134978 -1.01118715 0.4466501 -1.4881603 -0.4606660 0.1662918  
## 5 1.2408472 -1.3134978 0.09041516 0.4466501 -1.4881603 0.5839095 -1.9361058  
## 6 0.1205182 -1.3134978 0.09041516 -0.5531463 0.6772681 -0.4606660 -1.9361058  
## S29 S30 S31 S32 S33 S34 S35  
## 1 2.037201 0.3483267 -0.3073885 -0.98561979 1.3086619 1.6410617 -0.2650322  
## 2 1.024293 -1.7006720 1.7763453 0.09819406 -1.5893706 -1.4883974 -0.2650322  
## 3 -1.001523 0.3483267 -0.3073885 0.09819406 -1.5893706 -1.4883974 -0.2650322  
## 4 -1.001523 -1.7006720 -0.3073885 2.26582174 -0.6233597 0.5979086 -1.4365117  
## 5 1.024293 1.3728260 -0.3073885 -0.98561979 1.3086619 1.6410617 -0.2650322  
## 6 1.024293 -0.6761727 0.7344784 -0.98561979 0.3426511 0.5979086 0.9064473  
## S36 S37 S38 S39 S40 S41 S42  
## 1 -0.4452195 1.7382453 1.6053695 1.2589067 -1.4408345 -0.315485 1.4507537  
## 2 1.5673397 -1.3723401 -0.3976215 -0.9776499 1.8281697 -0.315485 1.4507537  
## 3 -0.4452195 -1.3723401 1.6053695 -0.9776499 1.8281697 -1.366256 0.3598699  
## 4 0.5610601 -0.3354783 -1.3991170 -0.9776499 -0.3511664 -0.315485 0.3598699  
## 5 0.5610601 0.7013835 0.6038740 1.2589067 -0.3511664 0.735286 -0.7310139  
## 6 1.5673397 -0.3354783 1.6053695 0.1406284 -0.3511664 -0.315485 -0.7310139  
## S43 S44 S45 S46 S47 S48 S49  
## 1 -1.1657036 -0.2995858 -0.7231952 1.07141735 -0.3595538 1.03215321 1.5360945  
## 2 -0.1199154 -0.2995858 1.3531911 0.05563217 0.7566629 -0.06382748 -1.3132644  
## 3 1.9716612 0.8363723 1.3531911 0.05563217 1.8728797 -1.15980817 -1.3132644  
## 4 -0.1199154 -1.4355440 0.3149980 0.05563217 0.7566629 1.03215321 1.5360945  
## 5 -1.1657036 0.8363723 0.3149980 1.07141735 -1.4757706 1.03215321 0.5863082  
## 6 -0.1199154 -0.2995858 -1.7613884 0.05563217 -0.3595538 -0.06382748 -1.3132644  
## S50 S51 S52 S53 S54 S55 S56  
## 1 -0.7815061 -1.5740365 1.6854243 -0.5550375 1.6289016 0.4235176 -0.4226726  
## 2 0.2835340 0.4057934 0.6531282 1.5769755 -0.3525363 0.4235176 1.7343813  
## 3 0.2835340 1.3957084 0.6531282 -1.6210440 0.6381827 -1.7820015 0.6558544  
## 4 -0.7815061 0.4057934 -0.3791679 0.5109690 -0.3525363 0.4235176 -0.4226726  
## 5 1.3485741 -0.5841215 -1.4114640 1.5769755 -1.3432553 0.4235176 -0.4226726  
## 6 0.2835340 0.4057934 -0.3791679 0.5109690 -0.3525363 0.4235176 0.6558544  
## S57 S58 S59 S60  
## 1 -1.4242136 0.9385944 1.069148 0.4120159  
## 2 1.8729180 -0.1728946 1.069148 -0.6330823  
## 3 1.8729180 -0.1728946 -1.385624 1.4571141  
## 4 0.7738741 -0.1728946 1.069148 -1.6781805  
## 5 -1.4242136 -0.1728946 1.069148 -1.6781805  
## 6 -0.3251697 0.9385944 1.069148 -0.6330823

##Exploratory Data Analysis

Looking at our EQ score:

data %>% select("EQ") %>% ggplot(aes(x = EQ)) + geom\_histogram() + labs(title = "EQ distribution", x = "EQ Score", y = "Count")

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

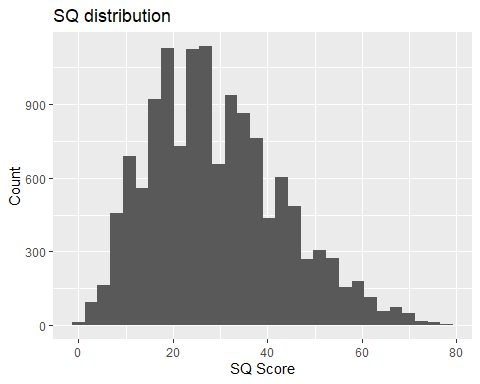


We see it’s a fairly symmetric looking distribution, with most people having between 20 and 60 as their EQ score.

Observing our SQ:

data %>% select("SQ") %>% ggplot(aes(x = SQ)) + geom\_histogram() + labs(title = "SQ distribution", x = "SQ Score", y = "Count")

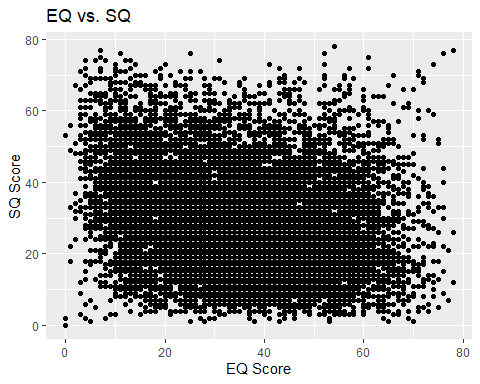
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



We see that the data is more left skewed here, with more people between 10 and 30 as their score, and a few high score individuals above 60.

Let’s see if there’s any relationship between EQ and SQ:

data %>% select("SQ", "EQ") %>% ggplot(aes(x = EQ, y = SQ)) + geom\_point() + labs(title = "EQ vs. SQ", x = "EQ Score", y = "SQ Score")



None in particular: people seem to be all over the map. Some of this could be due to the lack of regulation when it comes to the open psychometrics test takers - there could be those who are contrary, or answering in a tired way, thus adding much more noise.

Regardless, now that we understand our data a bit more, let us conduct a PCA.

##Performing PCA

Principal Component Analysis is used to group together variables which are multicolinear with other. Multicolinearity is the issue when your supposedly independent variables aren’t actually independent of each other: there are significant correlations between them, and thus this biases the model.

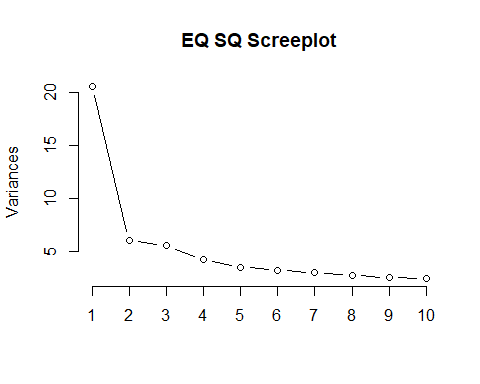
Traditionally, one removes one of the multicolinear variables, but this may take away from the model essential information. Thus, PCA aims to group similar variables together into a representative component variable that, although reduced in specificity, represents the group of variables that all inform the dependent variable.

This is especially useful in psychological questionnaires, because they tend to ask the same question in slightly different ways repeatedly, in order to discern people’s results more accurately; in our case with 120 questions where the user has to rate their response from 1 to 5, this is especially true.

The first question is - how many components do we need?

We solve this by utilizing a Screeplot. A screeplot conducts what’s known as an eigenvalue criterion in order to see how much benefit we get from each additional component: we generally continue to include components until we no longer get significant increases in the quality of our model based on the increased number of components.

pc.x = prcomp(X)  
  
plot(pc.x, type = "lines", main = "EQ SQ Screeplot")



In our case, we get diminishing returns at about components = 3. So, we will conduct a PCA with 3 components.

pcal1 = principal(X, rotate = "varimax", nfactors = 3)

## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was done

## The determinant of the smoothed correlation was zero.  
## This means the objective function is not defined.  
## Chi square is based upon observed residuals.

## The determinant of the smoothed correlation was zero.  
## This means the objective function is not defined for the null model either.  
## The Chi square is thus based upon observed correlations.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.18988e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.1844e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.1818e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.1571e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14475e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14444e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14429e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.1429e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.1422e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14218e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14202e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14201e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14197e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14195e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.18988e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.15737e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14965e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14232e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14195e+06, ..): not converged in 1000000  
## iter.

## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.  
  
## Warning in pchisq(df = result$dof, ncp = x, q = result$STATISTIC):  
## pnchisq(x=2.16337e+06, f=6783, theta=2.14194e+06, ..): not converged in 1000000  
## iter.

## Warning in principal(X, rotate = "varimax", nfactors = 3): The matrix is not  
## positive semi-definite, scores found from Structure loadings

print(pcal1$loadings, cutoff = .5)

##   
## Loadings:  
## RC1 RC3 RC2   
## E1   
## E2   
## E3   
## E4   
## E5 0.660   
## E6   
## E7 0.613   
## E8   
## E9   
## E10   
## E11 -0.535   
## E12   
## E13 0.619   
## E14   
## E15 0.512   
## E16 0.564  
## E17   
## E18 -0.593   
## E19 0.595   
## E20 0.642   
## E21   
## E22   
## E23   
## E24 -0.535   
## E25   
## E26   
## E27   
## E28   
## E29   
## E30   
## E31   
## E32   
## E33 0.655   
## E34 0.533   
## E35   
## E36   
## E37 0.500 0.523   
## E38   
## E39   
## E40 -0.548   
## E41 0.503   
## E42   
## E43   
## E44   
## E45   
## E46   
## E47   
## E48 0.593   
## E49 0.576   
## E50   
## E51   
## E52 0.503  
## E53 0.697   
## E54 0.541  
## E55 0.635   
## E56   
## E57 -0.606   
## E58   
## E59   
## E60 -0.652   
## S1   
## S2   
## S3   
## S4   
## S5 0.660   
## S6   
## S7 0.613   
## S8   
## S9   
## S10   
## S11 -0.535   
## S12   
## S13 0.619   
## S14   
## S15 0.512   
## S16 0.564  
## S17   
## S18 -0.593   
## S19 0.595   
## S20 0.642   
## S21   
## S22   
## S23   
## S24 -0.535   
## S25   
## S26   
## S27   
## S28   
## S29   
## S30   
## S31   
## S32   
## S33 0.655   
## S34 0.533   
## S35   
## S36   
## S37 0.500 0.523   
## S38   
## S39   
## S40 -0.548   
## S41 0.503   
## S42   
## S43   
## S44   
## S45   
## S46   
## S47   
## S48 0.593   
## S49 0.576   
## S50   
## S51   
## S52 0.503  
## S53 0.697   
## S54 0.541  
## S55 0.635   
## S56   
## S57 -0.606   
## S58   
## S59   
## S60 -0.652   
##   
## RC1 RC3 RC2  
## SS loadings 14.595 11.600 5.881  
## Proportion Var 0.122 0.097 0.049  
## Cumulative Var 0.122 0.218 0.267

Group 1 contains questions like “I am fascinated with how machines work” and “I can’t relax until I’ve done everything I planned to do today” to indicate a more orderly and systematic thinker who requires rigid rules to function well.

Group 2 contains questions like “I often forget the precise details of conversations I’ve had” and “I find it difficult to remember people’s faces” in order to indicate an individual who, although may be prosocial, is fairly forgetful.

Finally, group 3 contains questions like “I can tell if someone is masking their true emotion” and “I can usually appreciate the other’s viewpoint, even if I don’t agree with it”, to suggest someone who’s fairly emotionally observant and fair.

Let’s see how well these three components do in predicting EQ:

RC1 = pcal1$score[,1]   
RC2 = pcal1$score[,2]  
RC3 = pcal1$score[,3]  
  
#make 2 linear models, one for predicting SQ, EQ  
#predicting EQ - not very good at it  
eq = data$EQ  
fit2 = lm(eq ~ RC1 + RC2 + RC3)  
summary(fit2)

##   
## Call:  
## lm(formula = eq ~ RC1 + RC2 + RC3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -70.106 -9.089 -0.763 8.445 51.000   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 34.96077 0.11080 315.54 <2e-16 \*\*\*  
## RC1 -0.57439 0.01321 -43.47 <2e-16 \*\*\*  
## RC2 0.34936 0.01567 22.30 <2e-16 \*\*\*  
## RC3 -1.08536 0.01892 -57.35 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 12.76 on 13252 degrees of freedom  
## Multiple R-squared: 0.2867, Adjusted R-squared: 0.2866   
## F-statistic: 1776 on 3 and 13252 DF, p-value: < 2.2e-16

With all three components significant, but only RC2 having a positive correlation with EQ and an R^2 value of 28%, we may need to investigate further before we consider this the strongest possible model to predict EQ. However, it’s reasonable to consider that because EQ contains a greater bundle of skills than SQ, that it’s a more vaguely defined term and thus harder to predict.

Now, to see how our model does in predicting SQ:

#predicting SQ - actually really good at predicting this  
sq = data$SQ  
fit1 = lm(sq ~ RC1 + RC2 + RC3)  
summary(fit1)

##   
## Call:  
## lm(formula = sq ~ RC1 + RC2 + RC3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -22.1874 -3.1784 -0.2225 2.9545 18.1265   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 29.262296 0.039546 739.960 <2e-16 \*\*\*  
## RC1 0.534649 0.004716 113.364 <2e-16 \*\*\*  
## RC2 0.367472 0.005592 65.713 <2e-16 \*\*\*  
## RC3 0.016048 0.006754 2.376 0.0175 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.553 on 13252 degrees of freedom  
## Multiple R-squared: 0.8952, Adjusted R-squared: 0.8951   
## F-statistic: 3.771e+04 on 3 and 13252 DF, p-value: < 2.2e-16

The model does significantly better at predicting SQ than EQ (likely due to SQ’s clearer definition) though component 3 is less important and only minimally significant in this case. Our R^2 is nearly 90%, and our first two components are highly significant, suggesting that this is a fairly viable model for predicting an individual’s SQ score.

##Conclusion

Ultimately, PCA is a powerful data reduction tool to incorporate multicolinear but useful information into a linear model, especially useful in psychological surveys.

In our case, we investigated the EQ/SQ scaled tests, finding a linear regression after PCA that predicts SQ well, but doesn’t quite predict EQ well. This may be because of the difference in what SQ means vs. EQ: EQ is a variety of skills utilized in social interaction and self regulation, while SQ is a few interlocked skills in logical thinking.

Further study could be figuring out how to clean the data better - it’s clear that this data contains a fair amount of noise, though there is no clear and effective way to get rid of it. This is likely best done by observing other studies on this matter and developing a heuristic to remove the most extreme of data points, but regardless, it’s outside the scope of this study.