Aim 2

Ryan Canfield

2023-11-19

### This is the code for the preliminary results for completing Aim 2.

### Other code will be added later which will include the final model with the highest accuracy.

## By Ryan Canfield

## DSE4900 Data Scienc Capstone

## Dr. Santa Barabara Group (Ryan Canfield, Pat Norcross, and Alex Buterra)

## 11/14/23

# Libraries and Packages  
library(factoextra) # Used for PCA

## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

library(FactoMineR)

# Reading in the data set and previewing the data  
df <- read.csv("combined\_dataset.csv")

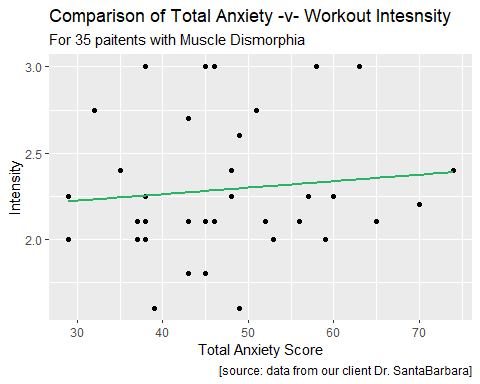
## Preprocessing   
# Dropping columns  
df2 = subset(df, select = -c(X, index, GROUP, ELIGIBILITYRESCREENQ1, ELIGIBILITYRESCREENQ2, ELIGIBILITYRESCREENQ3, ELIGIBILITYRESCREENQ4, GENDER2, RACEOTHER, LIVING, NUMBEROFROOMMATES, INCOME, EMPLOYMENT, HEALTHHIX23, DRUGADDICTION, ALCOHOLADDICTION, SU15))  
  
## THIS WAS DONE BUT AFTER DOING WORK THIS IS NOT NECCSARY BECAUSE THESE COLUMNS ARE NOT USED.  
# Binary column race into 1 for white all other for 0   
# living 0 alone 1 other people  
df2$RACE <- ifelse(df2$RACE != 5, 0, 1)  
df2$EDUCATION <- ifelse(df2$EDUCATION <= 5, 0, 1)  
  
# Fixes the Na's found in the age column  
df2$AGE2[is.na(df2$AGE2)] <- round(median(df2$AGE2, na.rm = TRUE))  
# Keep HISPANICLATINO column(if they are part or not )

# Preliminary model to see if we can use total scores to predict intensity   
Starting.Intensity <- lm(RTINTENSITY ~ MDISCORE + RSSCORE + BDSSCORE + SPASSCORE + CESDSCORE + STAISCORE + AUDITSCORE, data = df2)  
summary.lm(Starting.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ MDISCORE + RSSCORE + BDSSCORE + SPASSCORE +   
## CESDSCORE + STAISCORE + AUDITSCORE, data = df2)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.74425 -0.22155 0.00537 0.19124 0.73649   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.7956055 0.5002892 3.589 0.00135 \*\*  
## MDISCORE -0.0216740 0.0126963 -1.707 0.09972 .   
## RSSCORE 0.0024582 0.0148624 0.165 0.86991   
## BDSSCORE -0.0004274 0.0110825 -0.039 0.96953   
## SPASSCORE 0.0114813 0.0112930 1.017 0.31868   
## CESDSCORE -0.0370075 0.0192625 -1.921 0.06573 .   
## STAISCORE 0.0204297 0.0097660 2.092 0.04636 \*   
## AUDITSCORE 0.0265404 0.0275641 0.963 0.34449   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3748 on 26 degrees of freedom  
## (1 observation deleted due to missingness)  
## Multiple R-squared: 0.3195, Adjusted R-squared: 0.1363   
## F-statistic: 1.744 on 7 and 26 DF, p-value: 0.1424

# Plotting the forward stepwise linear regression model based off of the most significant variable  
g4 <- ggplot(data = df2, aes(x = STAISCORE, y = RTINTENSITY)) +   
 geom\_point () +   
 labs(title = "Comparison of Total Anxiety -v- Workout Intesnsity",   
 subtitle ="For 35 paitents with Muscle Dismorphia",   
 x = "Total Anxiety Score",   
 y = "Intensity",   
 caption = "[source: data from our client Dr. SantaBarbara]") +   
 geom\_smooth(method = "lm", se = 0, colour = "#28B463")   
g4

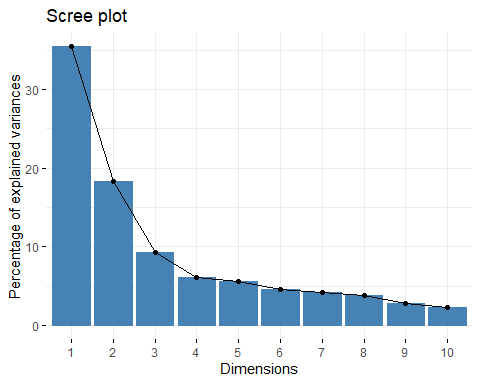
## `geom\_smooth()` using formula 'y ~ x'



## PCA for MDI  
# Just getting the MDI variables MDI1 - MDIScore  
subsetMDI <- df2[c(22:38)]  
  
set.seed(310) #I use this so that I get the same answer each time  
  
# Preforming PCA  
res.pca.MDI <- prcomp(subsetMDI[c(1:16)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
  
# Eigenvalues  
eig.val.MDI <- get\_eigenvalue(res.pca.MDI)  
eig.val.MDI

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 5.67195554 35.4497221 35.44972  
## Dim.2 2.93552572 18.3470358 53.79676  
## Dim.3 1.49064777 9.3165486 63.11331  
## Dim.4 0.96649179 6.0405737 69.15388  
## Dim.5 0.89025454 5.5640909 74.71797  
## Dim.6 0.72567076 4.5354423 79.25341  
## Dim.7 0.67331609 4.2082255 83.46164  
## Dim.8 0.61383354 3.8364596 87.29810  
## Dim.9 0.45277274 2.8298296 90.12793  
## Dim.10 0.35668608 2.2292880 92.35722  
## Dim.11 0.34707388 2.1692118 94.52643  
## Dim.12 0.27761166 1.7350729 96.26150  
## Dim.13 0.24933200 1.5583250 97.81983  
## Dim.14 0.13896788 0.8685493 98.68837  
## Dim.15 0.11282204 0.7051377 99.39351  
## Dim.16 0.09703797 0.6064873 100.00000

# Scree plot to look at which dimensions to keep  
fviz\_eig(res.pca.MDI)



# Looking at those dimensions to extract variables with high variance  
res.var.MDI <- get\_pca\_ind(res.pca.MDI)  
  
stored <- res.var.MDI$coord[, 1:3]  
# ^^^ Store this into a data frame  
  
MDIvari <- data.frame(stored)  
MDIvari

## Dim.1 Dim.2 Dim.3  
## 1 -1.38410884 0.023650609 -0.14737501  
## 2 1.65925125 1.691334730 -1.24816300  
## 3 0.28950627 0.807184630 -0.88596266  
## 4 1.24256582 -0.570013461 -0.30298414  
## 5 -2.04050490 1.944371928 0.25414048  
## 6 -3.85250219 0.483691542 1.18174003  
## 7 0.51852030 -1.403746590 -2.88561967  
## 8 2.25925690 -0.183704648 0.95462236  
## 9 5.36174223 -0.055384604 -0.15281568  
## 10 0.70200997 1.178939538 -0.70202407  
## 11 -2.17694020 -1.522025258 -0.20010306  
## 12 3.11911389 -1.095840086 -0.69717633  
## 13 -2.61215131 -2.383795590 0.98179146  
## 14 -0.82875627 1.729700636 1.02374339  
## 15 0.54791649 2.482111987 0.09555910  
## 16 -2.18950167 1.408111239 0.34854225  
## 17 2.42735064 0.143004158 0.42115643  
## 18 4.45273663 -0.060033302 -1.28144753  
## 19 0.77411780 -5.595989635 1.10011279  
## 20 -1.16034050 -0.210277190 1.23100336  
## 21 1.88597425 1.181716595 0.31710525  
## 22 3.55709113 -0.776537421 3.60144491  
## 23 -0.04252498 -3.077586950 -1.01888484  
## 24 -1.14275409 2.322841938 -1.10824307  
## 25 0.36495158 1.244704307 2.24915318  
## 26 -5.10150015 -2.019790836 -1.17601983  
## 27 1.39109485 -0.946118902 0.90077479  
## 28 -0.89268403 -1.167266273 -0.40908667  
## 29 -3.94589872 0.445951549 1.06646523  
## 30 1.23847892 -0.634239667 -0.65672497  
## 31 -3.07569438 -0.001113342 -1.07254598  
## 32 1.54061180 1.909936771 -1.41484530  
## 33 -0.96846840 2.797886369 0.83339549  
## 34 -1.14720010 0.005268376 0.07140566  
## 35 -0.77076000 -0.096943149 -1.27213433

# MDI6 MDI8 MDI9 MDI10  
# MDI2 MDI11 MDI13  
# MDI4

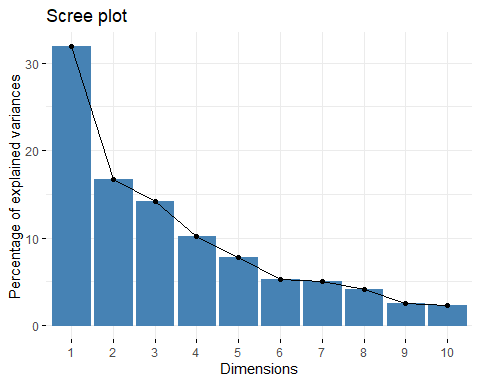
# This lets you look at what variables are used or questions ask  
which.var.MDI <- get\_pca\_var(res.pca.MDI)  
which.var.MDI$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## MDI1 0.6546385 -0.26492980 0.10038121 0.40215312 -0.17030950 0.09007370  
## MDI2 0.3526982 0.70435111 0.15177432 0.06786822 0.05746514 0.24020348  
## MDI3 0.5840656 -0.42087284 -0.20025636 -0.27213014 0.35463437 -0.08554338  
## MDI4 0.3188706 0.39890340 -0.70188976 0.10440428 0.23856953 -0.10308065  
## MDI5 0.6465778 -0.20891023 0.03616195 -0.44618986 -0.22964731 -0.30762820  
## MDI6 0.8254526 -0.20925353 -0.14269922 0.16185690 0.19050578 -0.10290030  
## MDI7 0.7136040 0.17236128 -0.34046962 0.18573782 -0.36450024 -0.20617695  
## MDI8 0.7557435 0.14739671 -0.23449464 -0.23687160 -0.20394631 -0.05301055  
## MDI9 0.7433029 -0.06994251 0.12397206 0.21359715 -0.27064028 0.27090991  
## MDI10 0.7141599 -0.44175949 0.23638416 0.17610413 0.11979220 -0.11522906  
## MDI11 0.3086273 0.74800724 0.39798697 0.03039218 0.12411404 -0.08697267  
## MDI12 0.5834782 0.45481944 0.06821596 -0.37953340 -0.26846002 0.27571603  
## MDI13 0.2859959 0.71337353 -0.29723150 0.01632549 0.28921480 0.11162151  
## MDI14 0.5936600 -0.38324593 -0.04127852 0.20295514 0.17915453 0.25185988  
## MDI15 0.4395705 0.43177154 0.57491562 0.12303607 0.16011883 -0.41963415  
## MDI16 0.6088627 -0.29236960 0.27808993 -0.34801275 0.30875584 0.26293494  
## Dim.7 Dim.8 Dim.9 Dim.10 Dim.11 Dim.12  
## MDI1 -0.340279630 0.205862619 -0.23879492 0.16487688 -0.11459828 0.10112937  
## MDI2 -0.348259263 0.092398992 0.07419502 0.03043592 0.36469634 0.05283271  
## MDI3 0.239600273 0.100374815 -0.17537670 0.24717582 0.14928275 0.21887517  
## MDI4 -0.083371441 -0.030348896 -0.26920515 -0.22372871 -0.06435263 0.02684719  
## MDI5 -0.303311149 -0.007451766 -0.05912500 0.13248803 -0.02534707 -0.17418594  
## MDI6 0.145526943 0.014002507 0.10793912 -0.06162043 0.23256882 -0.21874247  
## MDI7 0.002208647 -0.213134819 -0.01253423 -0.05583305 0.09974866 -0.01827950  
## MDI8 -0.027945306 0.009590262 0.38905597 -0.05777013 -0.09828049 0.27721319  
## MDI9 0.343764521 0.140049369 -0.06159074 -0.10975319 -0.03829679 0.05110847  
## MDI10 0.079491048 0.257559139 0.17426577 -0.04319325 -0.02133124 -0.10446151  
## MDI11 0.165887493 -0.249092889 -0.03150722 0.12887212 0.02116609 0.04468353  
## MDI12 0.229846234 -0.003183158 -0.20283678 0.02930553 -0.03614672 -0.13324863  
## MDI13 -0.014137189 0.256294525 0.16319591 0.17965598 -0.24922276 -0.14546812  
## MDI14 -0.072684434 -0.537918666 0.08905799 0.15388897 -0.12159830 -0.03447552  
## MDI15 0.021692846 -0.026516596 -0.08272685 -0.10539857 -0.12770982 0.06937421  
## MDI16 -0.217086169 -0.048948143 -0.05140768 -0.30195174 -0.05733525 0.01254563  
## Dim.13 Dim.14 Dim.15 Dim.16  
## MDI1 -0.1304744316 0.025845903 -0.006906647 0.09413956  
## MDI2 0.1113006430 -0.061530400 -0.017983275 -0.04802863  
## MDI3 -0.0237439187 -0.009093007 -0.016444245 -0.06362345  
## MDI4 0.1501722379 -0.055877806 0.064844775 0.05448281  
## MDI5 0.1988203719 0.077249911 0.027790255 -0.01315162  
## MDI6 -0.0611313764 0.092073874 -0.108776815 0.13635873  
## MDI7 -0.2248176050 0.011568472 0.034443749 -0.15627113  
## MDI8 -0.0002510846 -0.017265074 -0.021796248 0.09365643  
## MDI9 0.2331078332 0.135979032 -0.004113740 -0.06652342  
## MDI10 0.0176097408 -0.186173682 0.168807150 -0.02413593  
## MDI11 -0.0332267706 0.107802538 0.182643990 0.08516472  
## MDI12 -0.0877861218 -0.169259891 -0.046211808 0.05924258  
## MDI13 -0.0603334271 0.072742123 -0.027732151 -0.08256760  
## MDI14 0.1094024976 -0.076037721 -0.042618305 -0.02171616  
## MDI15 0.0331971363 -0.060898931 -0.161136391 -0.04393056  
## MDI16 -0.1510782869 0.090712415 0.033942231 -0.04363788

### This is the same code as abov just replaced with different survey data  
## Not the different subsets  
  
## PCA for RS  
subsetRS <- df2[c(39:49)]  
  
set.seed(310)   
res.pca.RS <- prcomp(subsetRS[c(1:10)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.RS <- get\_eigenvalue(res.pca.RS)  
eig.val.RS

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 3.1882929 31.882929 31.88293  
## Dim.2 1.6691343 16.691343 48.57427  
## Dim.3 1.4219997 14.219997 62.79427  
## Dim.4 1.0230147 10.230147 73.02442  
## Dim.5 0.7788125 7.788125 80.81254  
## Dim.6 0.5203806 5.203806 86.01635  
## Dim.7 0.5085023 5.085023 91.10137  
## Dim.8 0.4082066 4.082066 95.18344  
## Dim.9 0.2517061 2.517061 97.70050  
## Dim.10 0.2299503 2.299503 100.00000

fviz\_eig(res.pca.RS)



res.var.RS <- get\_pca\_ind(res.pca.RS)  
RSvari <- data.frame(res.var.RS$coord[, 1:4])  
  
# RS7 RS8   
# RS2 RS4  
# RS1

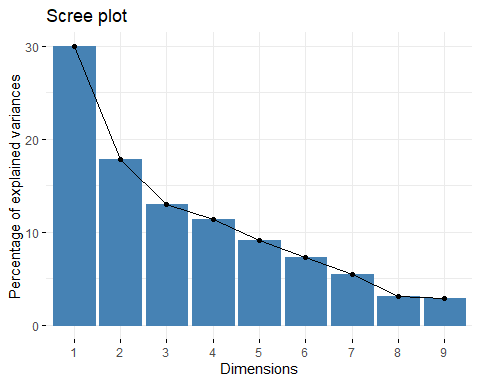
# This lets you look at what variables are used or questions ask  
which.var.RS <- get\_pca\_var(res.pca.RS)  
which.var.RS$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## RS1 0.4478273 -0.50466204 0.58958470 -0.1900479 0.15198291 -0.169772696  
## RS2 0.4342181 0.64306093 0.29244543 -0.2116156 0.26181154 -0.240576857  
## RS3 0.6295633 0.31538818 0.14092459 0.3320110 0.31925200 0.491719286  
## RS4 0.1658347 0.58941485 0.36526690 0.3420833 -0.55478145 -0.116700909  
## RS5 0.4963698 -0.04355438 -0.71126900 -0.2330657 -0.21332503 0.098257270  
## RS6 0.6638037 -0.17445998 -0.21907571 0.4354601 -0.18613445 -0.162721923  
## RS7 0.6856870 -0.43817968 -0.12415079 0.2294237 0.11277337 -0.228261933  
## RS8 0.8108417 0.03169587 -0.05999721 0.0710217 0.10576212 0.008776075  
## RS9 0.4560529 -0.39889956 0.46943936 -0.3198000 -0.41184990 0.294028518  
## RS10 0.5938970 0.41180511 -0.20567318 -0.5576438 -0.06153609 -0.059223278  
## Dim.7 Dim.8 Dim.9 Dim.10  
## RS1 0.05717810 -0.07032293 -0.31642724 -0.02786068  
## RS2 0.29095506 0.09457855 0.12016696 0.18209684  
## RS3 0.10510189 0.05080716 -0.07343758 -0.10657763  
## RS4 -0.11699790 0.14202011 -0.13635371 -0.02821349  
## RS5 0.17947050 0.18182064 -0.21665670 0.15532616  
## RS6 0.30525815 -0.36134610 0.07289909 -0.03341198  
## RS7 -0.08424211 0.40334278 0.13375703 -0.13148709  
## RS8 -0.49189602 -0.16363935 0.02320605 0.22879395  
## RS9 0.08283750 0.03924448 0.20481006 0.06186379  
## RS10 -0.12054251 -0.13012288 0.02287756 -0.29181302

## PCA for BDS  
subsetBDS <- df2[c(50:59)]  
set.seed(310)   
res.pca.BDS <- prcomp(subsetBDS[c(1:9)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.BDS <- get\_eigenvalue(res.pca.BDS)  
eig.val.BDS

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 2.6958122 29.953469 29.95347  
## Dim.2 1.6062123 17.846804 47.80027  
## Dim.3 1.1704348 13.004831 60.80510  
## Dim.4 1.0233328 11.370365 72.17547  
## Dim.5 0.8199016 9.110017 81.28549  
## Dim.6 0.6528312 7.253680 88.53917  
## Dim.7 0.4949973 5.499970 94.03914  
## Dim.8 0.2772361 3.080401 97.11954  
## Dim.9 0.2592417 2.880463 100.00000

fviz\_eig(res.pca.BDS)



res.var.BDS <- get\_pca\_ind(res.pca.BDS)  
BDSvari <- data.frame(res.var.BDS$coord[, 1:4])  
  
# BDS4 BDS5 BDS7 BDS3  
# Results for Variables - Prints component matrix

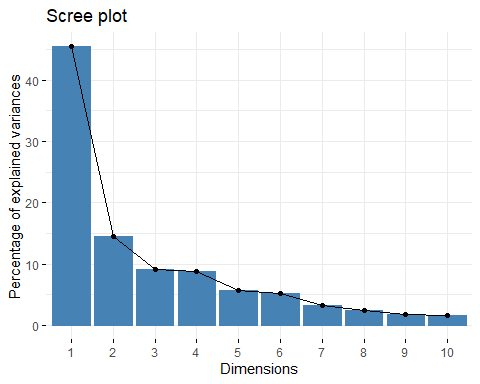
# This lets you look at what variables are used or questions ask  
which.var.BDS <- get\_pca\_var(res.pca.BDS)  
which.var.BDS$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## BDS1 0.5418330 -0.52669714 -0.01958913 0.13488816 -0.43086676 0.23646189  
## BDS2 0.4810704 -0.61094037 0.20617707 -0.05179112 -0.31375301 -0.09674402  
## BDS3 0.4398317 0.35436631 -0.26907397 -0.67377490 -0.06555280 0.27594733  
## BDS4 0.7537292 0.12799574 0.34635676 -0.17776780 0.34498052 0.06717995  
## BDS5 0.7520480 -0.25695136 0.17535036 0.10622758 0.43842675 0.10463239  
## BDS6 0.3047889 0.61996927 0.44448566 -0.14432107 -0.45708212 -0.14095157  
## BDS7 0.2266417 0.54401010 0.09042682 0.65281340 -0.07864162 0.38831041  
## BDS8 0.6872374 0.24635806 -0.19110251 0.22262856 0.01978744 -0.56957904  
## BDS9 0.4765570 0.08106008 -0.81373573 0.09507888 -0.06919247 0.02961348  
## Dim.7 Dim.8 Dim.9  
## BDS1 -0.37728668 -0.08174001 0.14085979  
## BDS2 0.48027901 0.10103467 -0.03815138  
## BDS3 0.17272105 -0.18469892 0.10103865  
## BDS4 -0.11029524 0.32030606 0.16018512  
## BDS5 -0.03217644 -0.22564760 -0.26692623  
## BDS6 -0.14658232 0.02245673 -0.23143824  
## BDS7 0.23553533 -0.01784466 0.07469552  
## BDS8 0.02136756 -0.15043297 0.18174265  
## BDS9 -0.03913258 0.22196258 -0.19660501

# PCA for SPAS  
subsetSPAS <- df2[c(60:72)]  
set.seed(310)   
res.pca.SPAS <- prcomp(subsetSPAS[c(1:12)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.SPAS <- get\_eigenvalue(res.pca.SPAS)  
eig.val.SPAS

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 5.4603398 45.5028317 45.50283  
## Dim.2 1.7329538 14.4412815 59.94411  
## Dim.3 1.0985428 9.1545235 69.09864  
## Dim.4 1.0567376 8.8061469 77.90478  
## Dim.5 0.6906914 5.7557620 83.66055  
## Dim.6 0.6302433 5.2520279 88.91257  
## Dim.7 0.3843170 3.2026420 92.11522  
## Dim.8 0.2921415 2.4345121 94.54973  
## Dim.9 0.2251598 1.8763320 96.42606  
## Dim.10 0.1863853 1.5532105 97.97927  
## Dim.11 0.1297895 1.0815790 99.06085  
## Dim.12 0.1126981 0.9391511 100.00000

fviz\_eig(res.pca.SPAS)



res.var.SPAS <- get\_pca\_ind(res.pca.SPAS)  
SPASvari <- data.frame(res.var.SPAS$coord[, 1:4])  
  
# SPAS1 SPAS2 SPAS4 SPAS6 SPAS8

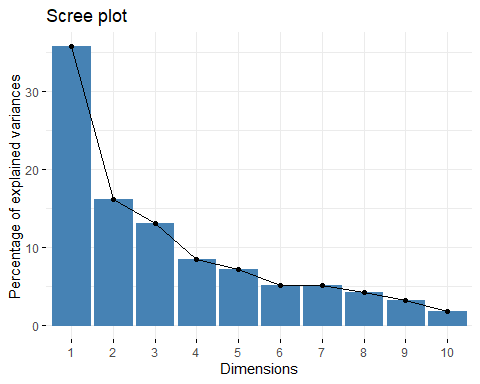
# This lets you look at what variables are used or questions ask  
which.var.SPAS <- get\_pca\_var(res.pca.SPAS)  
which.var.SPAS$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## SPAS1 0.7995385 -0.03125597 0.43321617 -0.06507578 0.05243399 -0.257804705  
## SPAS2 0.6264375 -0.38835205 -0.28885834 0.30328049 -0.39079346 0.103104684  
## SPAS3 0.6594829 0.40812596 0.10717102 0.04615238 -0.13219774 0.562491016  
## SPAS4 0.8532855 0.15598036 -0.03741986 0.18396652 0.27122596 -0.067625329  
## SPAS5 0.6204550 -0.29359045 0.57624913 -0.19114999 0.16324436 0.061012754  
## SPAS6 0.7518846 0.35671908 0.15911050 -0.28281849 -0.21222733 -0.023460982  
## SPAS7 0.5371318 0.51424128 -0.39845687 -0.31422443 0.19486494 -0.201183864  
## SPAS8 0.1131617 -0.60625589 -0.32023891 -0.54433662 0.32216903 0.310439999  
## SPAS9 0.6316833 0.17915581 -0.27273893 0.50075224 0.36274342 0.036011543  
## SPAS10 0.7083856 -0.38631737 -0.30231404 -0.16515674 -0.24683480 -0.289119085  
## SPAS11 0.6607047 -0.58514993 0.10459909 0.30703436 0.09436416 -0.002325468  
## SPAS12 0.8225962 0.14522522 -0.13827597 -0.20170862 -0.17121223 0.078531359  
## Dim.7 Dim.8 Dim.9 Dim.10 Dim.11  
## SPAS1 0.067468673 -0.036084225 0.159119341 -0.191312711 -0.02787155  
## SPAS2 -0.238329614 0.135305346 -0.037880517 -0.202711022 0.01669260  
## SPAS3 -0.067114612 -0.001136248 0.132664571 0.132516965 -0.02030703  
## SPAS4 -0.182085931 -0.191700673 0.045687997 0.012484085 0.23974996  
## SPAS5 -0.080515297 0.319431511 -0.095155501 0.066903267 0.03360134  
## SPAS6 -0.036195741 -0.211488749 -0.326576121 -0.025746770 -0.05626365  
## SPAS7 -0.247498764 0.133577130 0.098743243 0.001490396 -0.14252629  
## SPAS8 0.008315056 -0.085683408 -0.037739662 -0.088402657 0.01391791  
## SPAS9 0.228567679 0.132645764 -0.182273264 -0.011228867 -0.03755115  
## SPAS10 0.070693030 0.049223898 -0.002357485 0.249950280 0.06615945  
## SPAS11 -0.026531647 -0.207044777 0.070601913 0.096206886 -0.19832091  
## SPAS12 0.397796581 0.030781511 0.116721397 -0.078505768 0.03035041  
## Dim.12  
## SPAS1 -0.173456440  
## SPAS2 0.008746878  
## SPAS3 -0.104636144  
## SPAS4 0.067569484  
## SPAS5 0.081784328  
## SPAS6 0.003117384  
## SPAS7 0.043915013  
## SPAS8 -0.061507977  
## SPAS9 -0.079137288  
## SPAS10 -0.107157074  
## SPAS11 0.099019054  
## SPAS12 0.164498790

# PCA for CESD  
subsetCESD <- df2[c(73:83)]  
set.seed(310)   
res.pca.CESD <- prcomp(subsetCESD[c(1:10)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.CESD <- get\_eigenvalue(res.pca.CESD)  
eig.val.CESD

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 3.5755943 35.755943 35.75594  
## Dim.2 1.6169939 16.169939 51.92588  
## Dim.3 1.3110616 13.110616 65.03650  
## Dim.4 0.8400040 8.400040 73.43654  
## Dim.5 0.7138257 7.138257 80.57479  
## Dim.6 0.5159781 5.159781 85.73458  
## Dim.7 0.5092264 5.092264 90.82684  
## Dim.8 0.4185209 4.185209 95.01205  
## Dim.9 0.3203036 3.203036 98.21508  
## Dim.10 0.1784915 1.784915 100.00000

fviz\_eig(res.pca.CESD)



res.var.CESD <- get\_pca\_ind(res.pca.CESD)  
CESDvari <- data.frame(res.var.CESD$coord[, 1:3])  
  
# CESD3 CESD9 CESD6

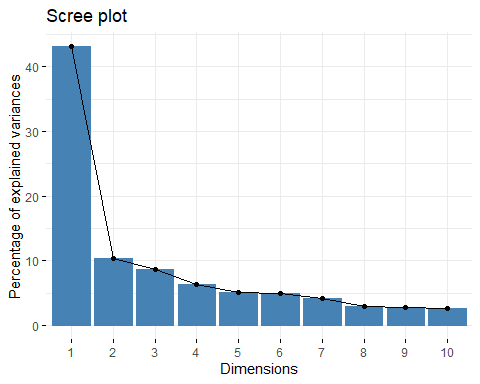
# This lets you look at what variables are used or questions ask  
which.var.CESD <- get\_pca\_var(res.pca.CESD)  
which.var.CESD$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## CESD1 0.6283022 -0.5541705 0.22261103 -0.02915794 0.16387663 -0.15660647  
## CESD2 0.5754787 -0.4032446 -0.14987117 0.57724279 0.03443592 -0.29019605  
## CESD3 0.6895893 0.3367244 0.10716950 -0.13374475 -0.34140383 -0.11095167  
## CESD4 0.6357702 -0.2002081 0.24180819 -0.49562369 -0.09443455 -0.28243062  
## CESD5 0.5668901 0.6257715 0.10636890 0.36625293 0.10993931 -0.04366911  
## CESD6 0.4655829 -0.2452593 -0.69535041 0.04523761 -0.12834475 0.16898217  
## CESD7 0.1288045 -0.4248721 0.74505590 0.22947937 -0.14516514 0.37104428  
## CESD8 0.6317297 0.4821213 0.19696831 -0.05429612 0.46999024 0.08757758  
## CESD9 0.7481364 0.2295175 -0.07045827 0.02725518 -0.43313553 0.20598724  
## CESD10 0.6755775 -0.2764751 -0.27457670 -0.22332647 0.31965708 0.31132795  
## Dim.7 Dim.8 Dim.9 Dim.10  
## CESD1 0.24613830 0.007279459 0.349905550 0.115210340  
## CESD2 -0.08259400 -0.135671831 -0.146020320 -0.136372278  
## CESD3 0.41417981 -0.180775046 -0.208331067 0.072229784  
## CESD4 -0.32315018 0.207680245 -0.120657062 -0.028262599  
## CESD5 -0.21686460 0.141433153 0.013514981 0.245723725  
## CESD6 0.16013659 0.404066919 -0.056803340 0.018816475  
## CESD7 0.01488292 0.128390896 -0.140184521 0.004448956  
## CESD8 0.17453487 0.127120144 -0.002245614 -0.227034046  
## CESD9 -0.19910357 -0.124582896 0.280788843 -0.133655312  
## CESD10 -0.15116037 -0.304651068 -0.129148778 0.102188995

# PCA for STAI  
subsetSTAI <- df2[c(84:104)]  
set.seed(310)   
res.pca.STAI <- prcomp(subsetSTAI[c(1:20)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.STAI <- get\_eigenvalue(res.pca.STAI)  
eig.val.STAI

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 8.62069418 43.1034709 43.10347  
## Dim.2 2.05837608 10.2918804 53.39535  
## Dim.3 1.72168518 8.6084259 62.00378  
## Dim.4 1.27892121 6.3946061 68.39838  
## Dim.5 1.01524963 5.0762482 73.47463  
## Dim.6 0.98022449 4.9011224 78.37575  
## Dim.7 0.82210049 4.1105024 82.48626  
## Dim.8 0.59615316 2.9807658 85.46702  
## Dim.9 0.56131038 2.8065519 88.27357  
## Dim.10 0.52340075 2.6170037 90.89058  
## Dim.11 0.47987555 2.3993778 93.28996  
## Dim.12 0.29766152 1.4883076 94.77826  
## Dim.13 0.24521930 1.2260965 96.00436  
## Dim.14 0.21463247 1.0731623 97.07752  
## Dim.15 0.17439880 0.8719940 97.94952  
## Dim.16 0.14098261 0.7049131 98.65443  
## Dim.17 0.11901464 0.5950732 99.24950  
## Dim.18 0.08268580 0.4134290 99.66293  
## Dim.19 0.03877580 0.1938790 99.85681  
## Dim.20 0.02863795 0.1431897 100.00000

fviz\_eig(res.pca.STAI)



res.var.STAI <- get\_pca\_ind(res.pca.STAI)  
STAIvari <- data.frame(res.var.STAI$coord[, 1:5])  
  
# STAI1 STAI4 STAI9 STAI10 STAI12 STAI16 STAI19

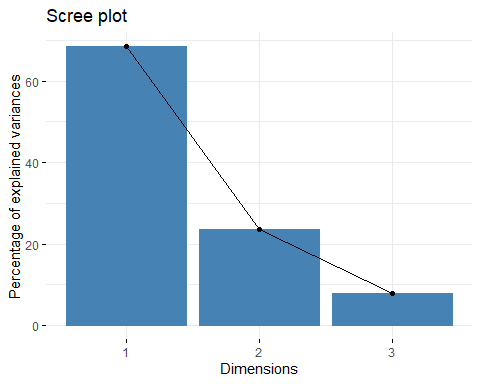
# This lets you look at what variables are used or questions ask  
which.var.STAI <- get\_pca\_var(res.pca.STAI)  
which.var.STAI$coord

## Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6  
## STAI1 0.8161973 -0.26329515 0.14524938 -0.16580201 0.17755371 -0.12373477  
## STAI2 0.7133062 -0.51472950 -0.08668850 0.29768300 0.08685232 0.20365346  
## STAI3 0.6628591 0.41750149 -0.15586096 0.05061835 0.35002390 0.13948337  
## STAI4 0.7491368 0.34232741 -0.04283788 -0.03406358 -0.28374357 -0.09137082  
## STAI5 0.3582702 0.65826504 0.25709148 0.24584272 -0.27088014 -0.01193882  
## STAI6 0.5430414 0.46100525 -0.10234472 -0.23582926 -0.05825342 0.33025647  
## STAI7 0.5944384 0.23827787 -0.28140739 -0.08059268 -0.41478317 -0.35162838  
## STAI8 0.6179407 -0.02626350 -0.11442824 0.51693859 -0.13473823 0.34889111  
## STAI9 0.6967516 -0.31980732 0.29675005 0.19280588 -0.07928609 -0.08069463  
## STAI10 0.7514830 0.04033683 0.02771973 -0.38781517 0.29714586 -0.11192181  
## STAI11 0.4926600 -0.19952337 0.57000571 -0.16230526 -0.09625657 -0.21203619  
## STAI12 0.7441146 0.09058826 0.16189631 0.30408966 0.14055400 -0.23883279  
## STAI13 0.8456009 0.06980039 0.01285836 -0.25692222 0.15722450 0.16350485  
## STAI14 0.4154086 -0.08394564 0.63915976 -0.15717040 -0.20033723 0.43776582  
## STAI15 0.6226042 0.28478750 0.16715546 0.43953243 0.18935912 -0.30842316  
## STAI16 0.8180366 0.14757085 -0.01586706 -0.13221499 0.11631080 0.22063719  
## STAI17 0.6060614 -0.57958932 0.09420214 -0.02636084 -0.22261708 -0.08471169  
## STAI18 0.4976272 -0.18881461 -0.45118926 -0.13213572 -0.44915623 0.07758876  
## STAI19 0.6889335 -0.03700335 -0.25372183 -0.32891507 0.03185331 -0.20880179  
## STAI20 0.6372248 -0.29940160 -0.58689100 0.18764117 0.12660257 0.03232073  
## Dim.7 Dim.8 Dim.9 Dim.10 Dim.11  
## STAI1 0.129139715 -0.077409603 -0.012457568 0.10512557 -0.259785835  
## STAI2 -0.073036659 0.008631135 -0.004238212 0.04016454 -0.029512837  
## STAI3 0.268928149 -0.036467065 -0.210872841 -0.06697185 -0.082608862  
## STAI4 0.177361559 0.047129559 0.051713000 -0.31308651 0.145001541  
## STAI5 -0.075912693 0.196573199 -0.311965946 0.06886251 -0.261342166  
## STAI6 -0.053977951 -0.401271627 0.187000207 0.28159410 -0.070949503  
## STAI7 0.256688131 -0.075605781 0.240048316 0.12614219 0.052766658  
## STAI8 0.068884795 0.148036188 0.248075903 -0.09322697 -0.159769153  
## STAI9 0.423509448 -0.057659254 -0.007381853 -0.09115962 0.023029798  
## STAI10 0.041524042 -0.111448943 -0.076423908 -0.15608701 -0.033869305  
## STAI11 -0.357982512 -0.020543722 0.235347100 -0.16452936 -0.252236366  
## STAI12 -0.204926641 -0.202305611 -0.046184303 -0.13546421 0.301118686  
## STAI13 -0.014533477 0.199128095 -0.104287351 0.02796875 0.147862620  
## STAI14 -0.004364453 0.089854873 0.017207287 0.14199336 0.296029198  
## STAI15 -0.243163978 -0.023312781 0.001299273 0.28553293 0.097847645  
## STAI16 -0.151711107 0.031645812 0.099198727 -0.17068538 -0.050617007  
## STAI17 0.164721832 -0.093135127 -0.263496855 0.20432686 -0.044957197  
## STAI18 -0.369998654 -0.137398810 -0.303939069 -0.12104114 0.004959073  
## STAI19 -0.075956664 0.474285587 0.087017628 0.17888542 0.037667031  
## STAI20 -0.116582473 0.019472632 0.093418254 0.03855329 0.030847283  
## Dim.12 Dim.13 Dim.14 Dim.15 Dim.16  
## STAI1 0.0702207533 -0.04021056 0.109804782 -0.034837715 -0.019554982  
## STAI2 0.1008902131 0.01013378 0.142218790 0.127891430 0.063319567  
## STAI3 -0.1026457142 0.16431584 -0.006812563 -0.179084012 -0.083298520  
## STAI4 0.0985886687 -0.01488098 -0.120855319 0.102724995 0.053650186  
## STAI5 0.0288756432 -0.10813158 -0.013974451 0.033133760 0.051925242  
## STAI6 0.0064520718 0.02245776 -0.123422102 0.019973245 0.116281165  
## STAI7 -0.0341951911 0.01919622 0.215558190 -0.016345848 -0.072501830  
## STAI8 -0.2394677802 0.04277630 0.001501680 0.065240957 -0.048134857  
## STAI9 0.1158478523 -0.12386891 -0.062638274 -0.147670737 0.092673464  
## STAI10 -0.2028486567 -0.23389447 -0.011051970 0.170028457 -0.075921488  
## STAI11 -0.0950914334 0.07341751 -0.024406292 -0.101917233 0.046157049  
## STAI12 -0.0875473528 0.14682043 -0.022291643 0.006739479 0.034185824  
## STAI13 -0.0480968297 0.05331397 0.185053054 0.008153520 0.199614005  
## STAI14 -0.0630020715 -0.10260818 0.005692497 -0.074039476 -0.121588353  
## STAI15 0.0602794788 -0.08285418 0.012756299 0.008694578 -0.070815364  
## STAI16 0.3453577281 0.07891597 0.005288625 0.034860922 -0.145643169  
## STAI17 -0.0277820077 0.17925577 -0.153611344 0.126016587 -0.049818222  
## STAI18 -0.0335060760 -0.03651726 0.066206002 -0.105326435 -0.036915938  
## STAI19 -0.0234059290 0.07712487 -0.142692777 -0.023774922 -0.001787498  
## STAI20 -0.0007773606 -0.19750754 -0.133618712 -0.112507378 0.025043802  
## Dim.17 Dim.18 Dim.19 Dim.20  
## STAI1 0.150971726 -0.151549873 -0.0184081965 0.037550325  
## STAI2 0.075335417 0.089034626 -0.0485833946 -0.079702131  
## STAI3 0.060183955 0.085661928 -0.0021880204 -0.022301198  
## STAI4 0.186243358 0.025866687 0.0281805922 0.014921194  
## STAI5 -0.046502593 0.003117127 -0.0690153777 0.007557114  
## STAI6 -0.007262285 -0.018438669 0.0033158800 -0.026341092  
## STAI7 -0.052489702 0.053090028 -0.0378867673 0.005781746  
## STAI8 -0.032010262 -0.078317877 0.0424808855 0.011342349  
## STAI9 -0.123213410 -0.030524225 0.0349377195 -0.047237317  
## STAI10 -0.066752162 0.019949885 0.0023783640 -0.025486318  
## STAI11 0.010785645 0.098211496 0.0046595004 0.004266288  
## STAI12 -0.039163445 -0.098425039 -0.0798605743 0.000671099  
## STAI13 -0.059523118 0.018944108 0.0496475981 0.058268529  
## STAI14 0.056422527 0.010493772 -0.0330706997 0.004449589  
## STAI15 0.035746547 0.030650124 0.1019466670 -0.003093926  
## STAI16 -0.122821045 -0.007266535 -0.0006945137 0.021226867  
## STAI17 -0.041342848 0.050443057 0.0141935792 0.046213609  
## STAI18 0.011813585 -0.048609414 0.0482414152 -0.031076914  
## STAI19 -0.003306517 -0.057832904 -0.0125449301 -0.064674658  
## STAI20 0.002539363 0.064080544 -0.0547682245 0.071996122

set.seed(310)   
# PCA for AUDIT  
subsetAUDIT <- df2[c(115:118)]  
subsetAUDIT <- replace(subsetAUDIT, is.na(subsetAUDIT), 0)  
  
res.pca.AUDIT <- prcomp(subsetAUDIT[c(1:3)], scale = TRUE) # leave off the last variable, which is the response variable "quality"  
eig.val.AUDIT <- get\_eigenvalue(res.pca.AUDIT)  
eig.val.AUDIT

## eigenvalue variance.percent cumulative.variance.percent  
## Dim.1 2.0556602 68.522007 68.52201  
## Dim.2 0.7102612 23.675373 92.19738  
## Dim.3 0.2340786 7.802619 100.00000

fviz\_eig(res.pca.AUDIT)



res.var.AUDIT <- get\_pca\_ind(res.pca.AUDIT)  
  
AUDITvari <- data.frame(res.var.AUDIT$coord[, 1:1])  
  
# AUDIT1 AUDIT2 AUDIT3

# This lets you look at what variables are used or questions ask  
which.var.AUDIT <- get\_pca\_var(res.pca.AUDIT)  
which.var.AUDIT$coord

## Dim.1 Dim.2 Dim.3  
## AUDIT1 -0.7904074 0.56274930 -0.2420111  
## AUDIT2 -0.7531391 -0.62667348 -0.2001546  
## AUDIT3 -0.9293535 0.02923653 0.3680318

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

# Renaming the columns to tell the dimensions apart.  
MDIvari <- MDIvari %>% rename("MDI.Dim.1" = "Dim.1", "MDI.Dim.2" = "Dim.2", "MDI.Dim.3" = "Dim.3")  
RSvari <- RSvari %>% rename("RS.Dim.1" = "Dim.1", "RS.Dim.2" = "Dim.2", "RS.Dim.3" = "Dim.3", "RS.Dim.4" = "Dim.4")  
BDSvari <- BDSvari %>% rename("BDS.Dim.1" = "Dim.1", "BDS.Dim.2" = "Dim.2", "BDS.Dim.3" = "Dim.3", "BDS.Dim.4" = "Dim.4")  
SPASvari <- SPASvari %>% rename("SPAS.Dim.1" = "Dim.1", "SPAS.Dim.2" = "Dim.2", "SPAS.Dim.3" = "Dim.3", "SPAS.Dim.4" = "Dim.4",)  
CESDvari <- CESDvari %>% rename("CESD.Dim.1" = "Dim.1", "CESD.Dim.2" = "Dim.2", "CESD.Dim.3" = "Dim.3")  
STAIvari <- STAIvari %>% rename("STAI.Dim.1" = "Dim.1", "STAI.Dim.2" = "Dim.2", "STAI.Dim.3" = "Dim.3", "STAI.Dim.4" = "Dim.4", "STAI.Dim.5" = "Dim.5")  
AUDITvari <- AUDITvari %>% rename("AUDIT.Dim.1" = "res.var.AUDIT.coord...1.1.")

# Forming a new dataset with the new PCA dimensions and the workout variables.   
questionaire.Variables <- cbind(MDIvari, RSvari, BDSvari, SPASvari, CESDvari, STAIvari, AUDITvari)  
df2.1 <- df2[ , 19:21]  
  
# Here are the scores if they are needed.  
#questionaire.Scores <- cbind(subsetMDI[c(17)], subsetRS[c(11)], subsetBDS[c(10)], subsetSPAS[c(13)], subsetCESD[c(11)], subsetSTAI[c(21)], subsetAUDIT[c(4)])  
#questionaire.Scores  
  
# This is the dataset I am using for the analysis.  
df3 <- cbind(df2.1, questionaire.Variables)  
head(df3)

## DAYSPASTWEEK RTINTENSITY MINSPERWORKOUT MDI.Dim.1 MDI.Dim.2 MDI.Dim.3  
## 1 3 3.00 90 -1.3841088 0.02365061 -0.1473750  
## 2 4 1.60 93 1.6592513 1.69133473 -1.2481630  
## 3 4 1.60 42 0.2895063 0.80718463 -0.8859627  
## 4 3 2.25 69 1.2425658 -0.57001346 -0.3029841  
## 5 6 3.00 75 -2.0405049 1.94437193 0.2541405  
## 6 4 2.25 70 -3.8525022 0.48369154 1.1817400  
## RS.Dim.1 RS.Dim.2 RS.Dim.3 RS.Dim.4 BDS.Dim.1 BDS.Dim.2  
## 1 -0.9322719 0.4975657 -1.8267220 -1.1447993 0.3547363 0.82262091  
## 2 1.7548255 0.6781711 -1.5324663 2.0617650 2.1965005 -0.88645209  
## 3 -0.6423681 1.3917457 -1.5016735 0.7702108 1.6744458 0.08007147  
## 4 -0.7265224 -1.7992907 -0.8112396 0.2710800 -0.8199347 0.23498504  
## 5 -1.2608750 -0.1409728 -0.9234223 0.6251850 -0.9225997 -0.59131311  
## 6 -1.9581331 -0.3462269 -0.6923884 0.1812494 -2.8493685 1.36970884  
## BDS.Dim.3 BDS.Dim.4 SPAS.Dim.1 SPAS.Dim.2 SPAS.Dim.3 SPAS.Dim.4  
## 1 0.98299860 -1.0736022 0.8151626 -0.6962117 -1.0548675 0.3833825  
## 2 1.45331048 -2.1602171 2.2695522 0.9431669 -1.5232404 0.5065902  
## 3 -1.38753402 0.5595683 -2.1122545 -0.4324762 -0.2534792 0.4687368  
## 4 -0.08357584 1.0738647 -1.6529051 -1.0908972 0.3979995 -0.4862350  
## 5 -0.98695123 -0.8691890 -0.9364750 -0.5514442 2.0019516 0.3670644  
## 6 -0.68488917 -0.4993194 -2.4964439 -1.6842082 -0.1889900 -0.2953370  
## CESD.Dim.1 CESD.Dim.2 CESD.Dim.3 STAI.Dim.1 STAI.Dim.2 STAI.Dim.3  
## 1 -0.08550555 0.75205575 0.2261907 -2.35573864 -0.7738701 0.4427402  
## 2 -0.67948328 0.07724049 0.8282880 0.41611465 0.9496104 0.5681745  
## 3 -0.90835535 -0.08223049 0.6618402 -2.30579763 -1.7316124 0.4528386  
## 4 0.91968849 -1.13435892 -1.2319074 -0.01755798 0.7117658 -0.6426774  
## 5 -1.80919212 1.56009366 0.5621977 -0.35894026 0.2737679 -1.5700976  
## 6 -3.05636640 -0.07920345 -0.5141475 -5.07042979 -1.4373212 -0.8251798  
## STAI.Dim.4 STAI.Dim.5 AUDIT.Dim.1  
## 1 0.9405375 0.99416533 -0.6867434  
## 2 -1.7644512 0.02665307 2.1149006  
## 3 -0.5800571 -0.05368946 0.4714605  
## 4 -0.4390572 -0.67388837 -0.9008528  
## 5 -0.0684286 0.33090864 -2.5442929  
## 6 0.4963490 -0.87149781 0.9566967

## Linear Regression

# Preforming linear regression on all the surveys to see if any dimensions are significant.  
MDI.Intensity <- lm(RTINTENSITY ~ MDI.Dim.1 + MDI.Dim.2 + MDI.Dim.3, data = df3)  
summary.lm(MDI.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ MDI.Dim.1 + MDI.Dim.2 + MDI.Dim.3,   
## data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.60071 -0.20654 -0.06862 0.13108 0.84206   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.28714 0.06564 34.842 <2e-16 \*\*\*  
## MDI.Dim.1 -0.01290 0.02797 -0.461 0.6477   
## MDI.Dim.2 -0.08238 0.03887 -2.119 0.0422 \*   
## MDI.Dim.3 0.01829 0.05455 0.335 0.7397   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3883 on 31 degrees of freedom  
## Multiple R-squared: 0.1345, Adjusted R-squared: 0.05073   
## F-statistic: 1.606 on 3 and 31 DF, p-value: 0.2081

RS.Intensity <- lm(RTINTENSITY ~ RS.Dim.1 + RS.Dim.2 + RS.Dim.3 + RS.Dim.4, data = df3)  
summary.lm(RS.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ RS.Dim.1 + RS.Dim.2 + RS.Dim.3 + RS.Dim.4,   
## data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.54430 -0.22985 -0.09029 0.14356 0.89484   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.287143 0.061340 37.286 <2e-16 \*\*\*  
## RS.Dim.1 0.002736 0.034855 0.078 0.9380   
## RS.Dim.2 0.066925 0.048172 1.389 0.1750   
## RS.Dim.3 0.071164 0.052190 1.364 0.1829   
## RS.Dim.4 -0.165363 0.061532 -2.687 0.0116 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3629 on 30 degrees of freedom  
## Multiple R-squared: 0.2686, Adjusted R-squared: 0.1711   
## F-statistic: 2.754 on 4 and 30 DF, p-value: 0.0461

BDS.Intensity <- lm(RTINTENSITY ~ BDS.Dim.1 + BDS.Dim.2 + BDS.Dim.3 + BDS.Dim.4, data = df3)  
summary.lm(BDS.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ BDS.Dim.1 + BDS.Dim.2 + BDS.Dim.3 +   
## BDS.Dim.4, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.6598 -0.2371 -0.1164 0.2808 0.7543   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.28714 0.06694 34.168 <2e-16 \*\*\*  
## BDS.Dim.1 -0.05539 0.04136 -1.339 0.191   
## BDS.Dim.2 0.05823 0.05359 1.087 0.286   
## BDS.Dim.3 0.07435 0.06278 1.184 0.246   
## BDS.Dim.4 -0.01752 0.06714 -0.261 0.796   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.396 on 30 degrees of freedom  
## Multiple R-squared: 0.129, Adjusted R-squared: 0.01292   
## F-statistic: 1.111 on 4 and 30 DF, p-value: 0.3695

SPAS.Intensity <- lm(RTINTENSITY ~ SPAS.Dim.1 + SPAS.Dim.2 + SPAS.Dim.3 + SPAS.Dim.4, data = df3)  
summary.lm(SPAS.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ SPAS.Dim.1 + SPAS.Dim.2 + SPAS.Dim.3 +   
## SPAS.Dim.4, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.60408 -0.24567 -0.05995 0.28325 0.79292   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.28714 0.06641 34.440 <2e-16 \*\*\*  
## SPAS.Dim.1 0.02787 0.02883 0.966 0.3416   
## SPAS.Dim.2 -0.04213 0.05118 -0.823 0.4169   
## SPAS.Dim.3 0.11493 0.06429 1.788 0.0839 .   
## SPAS.Dim.4 -0.02836 0.06554 -0.433 0.6683   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3929 on 30 degrees of freedom  
## Multiple R-squared: 0.1427, Adjusted R-squared: 0.02844   
## F-statistic: 1.249 on 4 and 30 DF, p-value: 0.3117

CESD.Intensity <- lm(RTINTENSITY ~ CESD.Dim.1 + CESD.Dim.2 + CESD.Dim.3, data = df3)  
summary.lm(CESD.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ CESD.Dim.1 + CESD.Dim.2 + CESD.Dim.3,   
## data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.7242 -0.2090 -0.1307 0.3160 0.7394   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.2871429 0.0689007 33.195 <2e-16 \*\*\*  
## CESD.Dim.1 -0.0433390 0.0369695 -1.172 0.250   
## CESD.Dim.2 0.0202353 0.0549748 0.368 0.715   
## CESD.Dim.3 -0.0009341 0.0610529 -0.015 0.988   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4076 on 31 degrees of freedom  
## Multiple R-squared: 0.04645, Adjusted R-squared: -0.04583   
## F-statistic: 0.5033 on 3 and 31 DF, p-value: 0.6828

STAI.Intensity <- lm(RTINTENSITY ~ STAI.Dim.1 + STAI.Dim.2 + STAI.Dim.3 + STAI.Dim.4 + STAI.Dim.5, data = df3)  
summary.lm(STAI.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ STAI.Dim.1 + STAI.Dim.2 + STAI.Dim.3 +   
## STAI.Dim.4 + STAI.Dim.5, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.57238 -0.24223 -0.02701 0.16520 0.72351   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.287143 0.063173 36.204 <2e-16 \*\*\*  
## STAI.Dim.1 0.009708 0.021830 0.445 0.6598   
## STAI.Dim.2 0.090380 0.044675 2.023 0.0524 .   
## STAI.Dim.3 0.023096 0.048849 0.473 0.6399   
## STAI.Dim.4 0.123993 0.056677 2.188 0.0369 \*   
## STAI.Dim.5 0.038786 0.063612 0.610 0.5468   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3737 on 29 degrees of freedom  
## Multiple R-squared: 0.2501, Adjusted R-squared: 0.1208   
## F-statistic: 1.934 on 5 and 29 DF, p-value: 0.119

AUDIT.Intensity <- lm(RTINTENSITY ~ AUDIT.Dim.1, data = df3)  
summary.lm(AUDIT.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ AUDIT.Dim.1, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.66592 -0.26865 -0.09193 0.23408 0.78622   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.28714 0.06748 33.891 <2e-16 \*\*\*  
## AUDIT.Dim.1 -0.04502 0.04776 -0.943 0.353   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3992 on 33 degrees of freedom  
## Multiple R-squared: 0.02622, Adjusted R-squared: -0.003283   
## F-statistic: 0.8887 on 1 and 33 DF, p-value: 0.3527

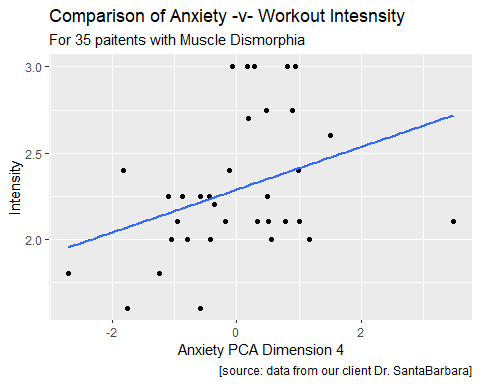
# Significant variables  
# MDI 2  
# RS 2 4  
# STAI 4

# Final linear regression model taken from the significant variables above  
FINAL.Intensity <- lm(RTINTENSITY ~ MDI.Dim.2 + RS.Dim.4 + STAI.Dim.4, data = df3)  
summary.lm(FINAL.Intensity)

##   
## Call:  
## lm(formula = RTINTENSITY ~ MDI.Dim.2 + RS.Dim.4 + STAI.Dim.4,   
## data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.53246 -0.19543 -0.04982 0.16110 0.87342   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.28714 0.05973 38.293 <2e-16 \*\*\*  
## MDI.Dim.2 -0.04708 0.04344 -1.084 0.2869   
## RS.Dim.4 -0.09871 0.07470 -1.321 0.1960   
## STAI.Dim.4 0.10688 0.05465 1.956 0.0596 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3534 on 31 degrees of freedom  
## Multiple R-squared: 0.2835, Adjusted R-squared: 0.2141   
## F-statistic: 4.088 on 3 and 31 DF, p-value: 0.01481

## Do scores first an see if anything is significant   
## Then go into PCA  
## TALK ABOUT WHAT QUESTIONS GO INTO THE THREE  
  
g1 <- ggplot(data = df3, aes(x = STAI.Dim.4, y = RTINTENSITY)) + # create cty-v-hwy graph from data set mpg  
 geom\_point () + # crate scatterplot with points jittered  
 labs(title = "Comparison of Anxiety -v- Workout Intesnsity", # create title above graph   
 subtitle ="For 35 paitents with Muscle Dismorphia", # create subtitle below title  
 x = "Anxiety PCA Dimension 4", # label x-axis  
 y = "Intensity", # label y-axis  
 caption = "[source: data from our client Dr. SantaBarbara]") # insert captionbelow graph # display graph  
  
g2 <- g1 + # create graph g2 starting with graph g1  
 geom\_smooth(method = "lm", se = 0) # add linear regression line with standard error envelope  
g2 # display graph

## `geom\_smooth()` using formula 'y ~ x'



#To run stepwise regrression:  
#First, let us define the null (intercept-only) model. We need this to build our forward stepwise regression:  
intercept\_only1 <- lm(DAYSPASTWEEK ~ 1, data = df3)  
summary.lm(intercept\_only1)

##   
## Call:  
## lm(formula = DAYSPASTWEEK ~ 1, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.8571 -0.8571 0.1429 0.1429 2.1429   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.8571 0.1647 23.42 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9745 on 34 degrees of freedom

#Next, let us define the model with all explanatory variables included. We need this both for forward and backward stepwise regression:  
  
all1 <- lm(RTINTENSITY ~ ., data = df3)  
summary.lm(all1)

##   
## Call:  
## lm(formula = RTINTENSITY ~ ., data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.4368 -0.1095 0.0289 0.1122 0.4068   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.012663 0.712932 1.420 0.1933   
## DAYSPASTWEEK 0.101658 0.122840 0.828 0.4319   
## MINSPERWORKOUT 0.012348 0.006416 1.925 0.0905 .  
## MDI.Dim.1 -0.025632 0.096303 -0.266 0.7968   
## MDI.Dim.2 -0.097142 0.174692 -0.556 0.5934   
## MDI.Dim.3 -0.063681 0.118990 -0.535 0.6071   
## RS.Dim.1 -0.064964 0.095920 -0.677 0.5173   
## RS.Dim.2 -0.085547 0.099725 -0.858 0.4159   
## RS.Dim.3 -0.042331 0.096422 -0.439 0.6723   
## RS.Dim.4 0.008745 0.233038 0.038 0.9710   
## BDS.Dim.1 -0.048197 0.078063 -0.617 0.5541   
## BDS.Dim.2 -0.069084 0.118463 -0.583 0.5758   
## BDS.Dim.3 -0.081997 0.099947 -0.820 0.4357   
## BDS.Dim.4 0.130393 0.134330 0.971 0.3601   
## SPAS.Dim.1 0.149451 0.105679 1.414 0.1950   
## SPAS.Dim.2 0.077958 0.101835 0.766 0.4659   
## SPAS.Dim.3 0.231406 0.165055 1.402 0.1985   
## SPAS.Dim.4 0.115577 0.112265 1.030 0.3334   
## CESD.Dim.1 -0.071251 0.111237 -0.641 0.5397   
## CESD.Dim.2 0.043770 0.116994 0.374 0.7180   
## CESD.Dim.3 -0.091046 0.113164 -0.805 0.4443   
## STAI.Dim.1 -0.014190 0.124338 -0.114 0.9120   
## STAI.Dim.2 -0.071757 0.121375 -0.591 0.5707   
## STAI.Dim.3 0.077024 0.097101 0.793 0.4505   
## STAI.Dim.4 0.114958 0.103458 1.111 0.2988   
## STAI.Dim.5 0.227800 0.137360 1.658 0.1358   
## AUDIT.Dim.1 0.018912 0.088726 0.213 0.8365   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3806 on 8 degrees of freedom  
## Multiple R-squared: 0.7854, Adjusted R-squared: 0.08805   
## F-statistic: 1.126 on 26 and 8 DF, p-value: 0.46

#Note the shortcut here: we can use the "." symbol instead of typing out all our variable names. This shortcut tells R to include all of them. We do have to tell R what dataset we are using (even though we have attached it).

#Perform forward stepwise regression here.  
#We use the step function.  
#We start building the model from the "intercept\_only" model.   
#We tell "step" the direction of stepwise regression we want.   
#scope=formula() tells "step" which predictors we would like to attempt to include in the model. Since our model "all" includes all the predictors, "step" will search all predictors to determine which ones work best in the optimal model.   
#trace = 0 suppresses the output of the step function for now.   
  
forward <- step(intercept\_only1, direction = 'forward', scope = formula(all1), trace = 0)

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 1 in model.matrix: no columns are assigned

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 2 in model.matrix: no columns are assigned

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 3 in model.matrix: no columns are assigned

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 4 in model.matrix: no columns are assigned

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 5 in model.matrix: no columns are assigned

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts): the  
## response appeared on the right-hand side and was dropped

## Warning in model.matrix.default(Terms, m, contrasts.arg = object$contrasts):  
## problem with term 6 in model.matrix: no columns are assigned

#If we want to see the output of the forward stepwise regression, we can use this command:  
forward$anova

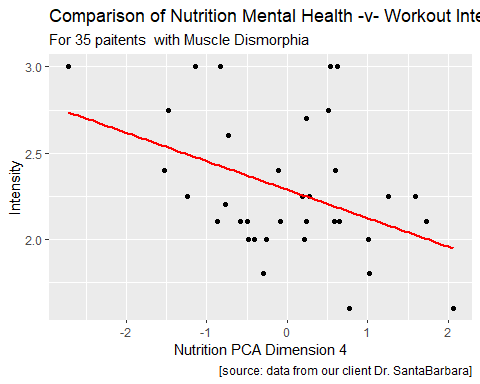
## Step Df Deviance Resid. Df Resid. Dev AIC  
## 1 NA NA 34 32.28571 -0.8253124  
## 2 + BDS.Dim.1 -1 3.095474 33 29.19024 -2.3529775  
## 3 + CESD.Dim.1 -1 2.272189 32 26.91805 -3.1892827  
## 4 + SPAS.Dim.3 -1 2.384340 31 24.53371 -4.4354973  
## 5 + STAI.Dim.4 -1 1.513360 30 23.02035 -4.6639298  
## 6 + CESD.Dim.2 -1 1.332288 29 21.68806 -4.7505127

# Note that forward$coefficients does not give us the full summary like summary.lm does. If we want that, we need to fit the model with these predictors and print out the output:  
bestforward <- lm(RTINTENSITY ~ RS.Dim.4 + MINSPERWORKOUT + SPAS.Dim.3 + STAI.Dim.5 + STAI.Dim.4 + STAI.Dim.2, data = df3)  
summary.lm(bestforward)

##   
## Call:  
## lm(formula = RTINTENSITY ~ RS.Dim.4 + MINSPERWORKOUT + SPAS.Dim.3 +   
## STAI.Dim.5 + STAI.Dim.4 + STAI.Dim.2, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.52084 -0.15089 0.00428 0.14328 0.48081   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.850912 0.164609 11.244 6.8e-12 \*\*\*  
## RS.Dim.4 -0.135159 0.050578 -2.672 0.0124 \*   
## MINSPERWORKOUT 0.006105 0.002199 2.776 0.0097 \*\*   
## SPAS.Dim.3 0.129030 0.048246 2.674 0.0124 \*   
## STAI.Dim.5 0.076519 0.050616 1.512 0.1418   
## STAI.Dim.4 0.076622 0.045761 1.674 0.1052   
## STAI.Dim.2 0.061909 0.036290 1.706 0.0991 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2899 on 28 degrees of freedom  
## Multiple R-squared: 0.5645, Adjusted R-squared: 0.4712   
## F-statistic: 6.049 on 6 and 28 DF, p-value: 0.0003741

# Plotting the forward stepwise linear regression model based off of the most significant variable  
g3 <- ggplot(data = df3, aes(x = RS.Dim.4, y = RTINTENSITY)) +   
 geom\_point () +   
 labs(title = "Comparison of Nutrition Mental Health -v- Workout Intesnsity",   
 subtitle ="For 35 paitents with Muscle Dismorphia",   
 x = "Nutrition PCA Dimension 4",   
 y = "Intensity",   
 caption = "[source: data from our client Dr. SantaBarbara]") +   
 geom\_smooth(method = "lm", se = 0, colour = "red")   
g3

## `geom\_smooth()` using formula 'y ~ x'



## Make regression trees

library(gbm)

## Warning: package 'gbm' was built under R version 4.2.3

## Loaded gbm 2.1.8.1

library(tree)

## Warning: package 'tree' was built under R version 4.2.3

library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

library(tidyverse)

## ── Attaching packages  
## ───────────────────────────────────────  
## tidyverse 1.3.2 ──

## ✔ tibble 3.1.8 ✔ purrr 0.3.4  
## ✔ tidyr 1.2.1 ✔ stringr 1.4.1  
## ✔ readr 2.1.3 ✔ forcats 0.5.2  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ randomForest::combine() masks dplyr::combine()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ randomForest::margin() masks ggplot2::margin()

library(BART)

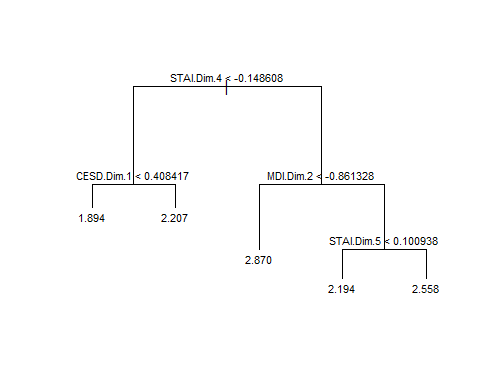
## Warning: package 'BART' was built under R version 4.2.3

## Loading required package: nlme  
##   
## Attaching package: 'nlme'  
##   
## The following object is masked from 'package:dplyr':  
##   
## collapse  
##   
## Loading required package: nnet  
## Loading required package: survival

head(df3)

## DAYSPASTWEEK RTINTENSITY MINSPERWORKOUT MDI.Dim.1 MDI.Dim.2 MDI.Dim.3  
## 1 3 3.00 90 -1.3841088 0.02365061 -0.1473750  
## 2 4 1.60 93 1.6592513 1.69133473 -1.2481630  
## 3 4 1.60 42 0.2895063 0.80718463 -0.8859627  
## 4 3 2.25 69 1.2425658 -0.57001346 -0.3029841  
## 5 6 3.00 75 -2.0405049 1.94437193 0.2541405  
## 6 4 2.25 70 -3.8525022 0.48369154 1.1817400  
## RS.Dim.1 RS.Dim.2 RS.Dim.3 RS.Dim.4 BDS.Dim.1 BDS.Dim.2  
## 1 -0.9322719 0.4975657 -1.8267220 -1.1447993 0.3547363 0.82262091  
## 2 1.7548255 0.6781711 -1.5324663 2.0617650 2.1965005 -0.88645209  
## 3 -0.6423681 1.3917457 -1.5016735 0.7702108 1.6744458 0.08007147  
## 4 -0.7265224 -1.7992907 -0.8112396 0.2710800 -0.8199347 0.23498504  
## 5 -1.2608750 -0.1409728 -0.9234223 0.6251850 -0.9225997 -0.59131311  
## 6 -1.9581331 -0.3462269 -0.6923884 0.1812494 -2.8493685 1.36970884  
## BDS.Dim.3 BDS.Dim.4 SPAS.Dim.1 SPAS.Dim.2 SPAS.Dim.3 SPAS.Dim.4  
## 1 0.98299860 -1.0736022 0.8151626 -0.6962117 -1.0548675 0.3833825  
## 2 1.45331048 -2.1602171 2.2695522 0.9431669 -1.5232404 0.5065902  
## 3 -1.38753402 0.5595683 -2.1122545 -0.4324762 -0.2534792 0.4687368  
## 4 -0.08357584 1.0738647 -1.6529051 -1.0908972 0.3979995 -0.4862350  
## 5 -0.98695123 -0.8691890 -0.9364750 -0.5514442 2.0019516 0.3670644  
## 6 -0.68488917 -0.4993194 -2.4964439 -1.6842082 -0.1889900 -0.2953370  
## CESD.Dim.1 CESD.Dim.2 CESD.Dim.3 STAI.Dim.1 STAI.Dim.2 STAI.Dim.3  
## 1 -0.08550555 0.75205575 0.2261907 -2.35573864 -0.7738701 0.4427402  
## 2 -0.67948328 0.07724049 0.8282880 0.41611465 0.9496104 0.5681745  
## 3 -0.90835535 -0.08223049 0.6618402 -2.30579763 -1.7316124 0.4528386  
## 4 0.91968849 -1.13435892 -1.2319074 -0.01755798 0.7117658 -0.6426774  
## 5 -1.80919212 1.56009366 0.5621977 -0.35894026 0.2737679 -1.5700976  
## 6 -3.05636640 -0.07920345 -0.5141475 -5.07042979 -1.4373212 -0.8251798  
## STAI.Dim.4 STAI.Dim.5 AUDIT.Dim.1  
## 1 0.9405375 0.99416533 -0.6867434  
## 2 -1.7644512 0.02665307 2.1149006  
## 3 -0.5800571 -0.05368946 0.4714605  
## 4 -0.4390572 -0.67388837 -0.9008528  
## 5 -0.0684286 0.33090864 -2.5442929  
## 6 0.4963490 -0.87149781 0.9566967

# Bottom of the node = the level of intensity   
tree\_model <- tree(RTINTENSITY ~ ., df3)  
  
plot(tree\_model)  
text(tree\_model, pretty = 0, cex = 0.7)



summary(tree\_model)

##   
## Regression tree:  
## tree(formula = RTINTENSITY ~ ., data = df3)  
## Variables actually used in tree construction:  
## [1] "STAI.Dim.4" "CESD.Dim.1" "MDI.Dim.2" "STAI.Dim.5"  
## Number of terminal nodes: 5   
## Residual mean deviance: 0.06339 = 1.902 / 30   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.458300 -0.139200 -0.007143 0.000000 0.130000 0.505600

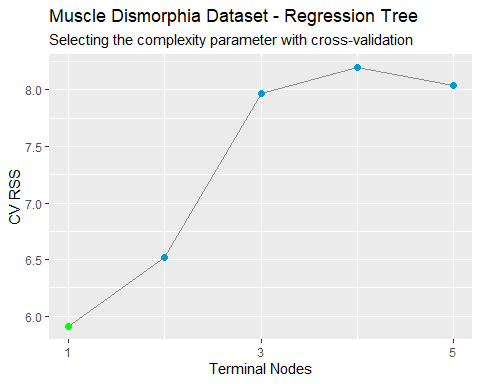
pred <- predict(tree\_model, df3)  
mean((pred - df3$RTINTENSITY)^2)

## [1] 0.05433245

#calculate residual standard error  
sqrt(deviance(tree\_model)/df.residual(tree\_model))

## numeric(0)

set.seed(310)  
  
cv\_tree\_model <- cv.tree(tree\_model, K = 10)  
  
data.frame(n\_leaves = cv\_tree\_model$size,  
 CV\_RSS = cv\_tree\_model$dev) %>%  
 mutate(min\_CV\_RSS = as.numeric(min(CV\_RSS) == CV\_RSS)) %>%  
 ggplot(aes(x = n\_leaves, y = CV\_RSS)) +  
 geom\_line(col = "grey55") +  
 geom\_point(size = 2, aes(col = factor(min\_CV\_RSS))) +  
 scale\_x\_continuous(breaks = seq(1, 17, 2)) +  
 scale\_y\_continuous(labels = scales::comma\_format()) +  
 scale\_color\_manual(values = c("deepskyblue3", "green")) +  
 theme(legend.position = "none") +  
 labs(title = "Muscle Dismorphia Dataset - Regression Tree",  
 subtitle = "Selecting the complexity parameter with cross-validation",  
 x = "Terminal Nodes",  
 y = "CV RSS")



# Seeing in pruning the tree helps.  
pruned\_tree\_model <- prune.tree(tree\_model, best = 2)  
cv.pred <- predict(pruned\_tree\_model, df3)  
mean((cv.pred - df3$RTINTENSITY)^2)

## [1] 0.108525

## Create table and talk about becarful of overfitting due to low sample size