UTSA PSY 2513 Disorders of Trauma, Stress, and Suicide

Augustine Osman / MITRE Collaboration

Fall 2019

Contents

Setting up your environment:

To perform the analysis without "reinventing the wheel", we will use pre-written code in R *packages*. To use a package, it must be *installed* once and then *loaded* into your workspace when you intend to use it.

This code assumes packages are already installed and simply need to be loaded by using the *library* or *require* function. If you have never installed them before, use *install.packages()* for each package. For example, *install.packages("dplyr")* will install the package, and *require(dplyr)* will load the package.

It is best practice to load all of your packages upfront to ensure your environment is ready for the analysis.

```
#Setting up your environment:

#requiring all packages
#this will load the packages given they have been installed.
require(dplyr)
require(reshape2)
require(tidyr)
require(ggridges)
require(kableExtra)
require(ggplot2)
require(lattice)
require(cluster)
require(gridExtra)
require(Hmisc)
require(corrplot)
require(stringr)
require(knitr)
require(psych)
require(rpart)
require(rpart.plot)
require(caTools)
library(caret)
library(e1071)
library(randomForest)

#setting our default options for how code chunks will appear
knitr::opts_chunk$set(error = FALSE,     # suppress errors
                      message = FALSE,   # suppress messages
                      warning = FALSE,   # suppress warnings
                      include = TRUE,    # do not suppress code output
                      echo = TRUE)       # do not suppress code
```

Data Ingest:

The first step of any data analysis is to properly ingest data of interest and store it in an appropriate way. In this case, we ingest a csv with 117 variables. To organize this data, we will split the variables into multiple data frames based on the category of the data. When splitting your data, it's important to keep a common key that can be used to join the data back together for different analyses.

For example, we will pull out each variable relating to *The Alcohol Use Disorders Test* and place those in a data frame called *Audit*, keeping the Study_ID from the original data set as key.

```
#reading in the data:
#note: set the file path to point wherever the data set of interest is stored
osmanB <- read.csv(file="osmanB.csv",
                   stringsAsFactors=FALSE, header = TRUE)
osmanB_cats <- read.csv(file="osmanB_cats.csv",
                        stringsAsFactors=FALSE, header = FALSE)

#creating a function to separate data based on category.
get_df <- function(source_nm){
  df_names<- as.character(osmanB_cats$V1[osmanB_cats$V2==source_nm])
  df_names <- c(df_names, 'Study_ID','Gender')
  new_df <- osmanB[,names(osmanB) %in% df_names]
  return(new_df)
}

#Get each data frame of interest using get_df():

#Get general information including ID, age, gender, race, ethnicity, combat, and service
osb_gen <- get_df('general')
#For a quick summary of what's in the general information data frame:
sumtable <- summary(osb_gen) #not printed, but now sumtable is in R environment to view

#Alcohol Use Disorders Test: self-report measure of problematic drinking for adults
Audit <- get_df('Audit')
#Measures related to Protective and Risk Factors: evaluate a sense of
#belongingness to one's family
MSRI <- get_df('MSRI')
#Reasons for Living: assess the reasons people give for not killing themselves
RFL <- get_df('RFL')
#Beck Scale for Suicide Ideation: measure of several factors related to suicidal ideation
BSS <- get_df('BSS')
```

Data Exploration:

First, we will get an overview of each of the data frames we created above. It is important to understand what type of data you have and what it means, which may be defined in a *data dictionary* accompanying your data set of interest.

The data frames created include:

- Alcohol Use Disorders Test (Audit): self-report measure of problematic drinking for adults
- Measures related to Protective and Risk Factors (MSRI): Suicide Ideation, Family Connectedness, Negative Affect, Positive Self-Perception
- Reasons for Living (RFL): assess the reasons people give for not killing themselves
- Beck Scale for Suicide Ideation (BSS): measure of several factors related to suicidal ideation
- General Information Table (osb_gen): general information including ID, age, gender, race, ethnicity, combat, and service for each observation

Data Overview

- Alcohol Use Disorders Test: self-report measure of problematic drinking for adults

```
#show the first 6 observations of all columns in the data set "Audit"
head(Audit)
```

```
##   Study_ID Gender Audit1 Audit2 Audit3 AUDITtot
## 1        6      2      3      1      1        5
## 2        8      1      5      1      3        9
## 3       12      2      1      1      1        3
## 4       17      2      1      1      1        3
## 5       20      2      2      3      2        7
## 6       21      2      5      2      3       10
```

- Measures related to Protective and Risk Factors: Suicide Ideation, Family Connectedness, Negative Affect, Positive Self-Perception

```
#show the first 6 observations of columns of interest in the data set "MSRI"
#Here, we specify columns by calling their names
head(MSRI[,c('Study_ID','Gender',
             'MSRIsui','MSRIfam','MSRIna', 'MSRIpse')])
```

```
##   Study_ID Gender MSRIsui MSRIfam MSRIna MSRIpse
## 1        6      2      16      14     19      17
## 2        8      1       8      31     27      28
## 3       12      2      11      18      7      31
## 4       17      2      28      13     35      11
## 5       20      2       7      15     13      27
## 6       21      2       9      17     20      19
```

- Reasons for Living: assess the reasons people give for not killing themselves *(48 variables; not all shown)*

```
#Show the first 6 observations of columns of interest in the data set "RFL"
#Here, we specify columns by calling the column numbers
head(RFL[,c(1,2,3:5,50:51)])
```

```
##   Study_ID Gender RFL1 RFL2 RFL3 RFL48 RFLtotal
## 1        6      2    5    3    3     5 3.229167
## 2        8      1    5    6    6     1 4.541667
## 3       12      2    1    6    6     2 3.291667
## 4       17      2    2    1    1     1 1.645833
## 5       20      2    6    6    5     1 4.354167
```

```
## 6       21      2    4    5    5     1 3.375000
```

- Beck Scale for Suicide Ideation: measure of several factors related to suicidal ideation *(22 variables; not all shown)*

```
#Show the first 6 observations of columns of interest in the data set "RFL"
#Here, we specify columns by calling the column numbers
head(BSS[,c(1,2,3:5,20:22)])
```

```
##   Study_ID Gender BSS1 BSS2 BSS3 BSS18 BSS19 BSS19_Total
## 1        6      2    0    1    0     0     1          10
## 2        8      1    0    0    0     0     0           0
## 3       12      2    0    0    0     0     0           2
## 4       17      2    1    1    1     0     1          14
## 5       20      2    0    0    0     0     0           0
## 6       21      2    0    0    0     0     0           0
```

Data Correlation

Exploring the relationship of risk and protective factors, regardless of combat group:

What variables are closely related? What is the connection, or the correlation, between variables? These questions can be explored using correlation matrices.

Risk Factors:

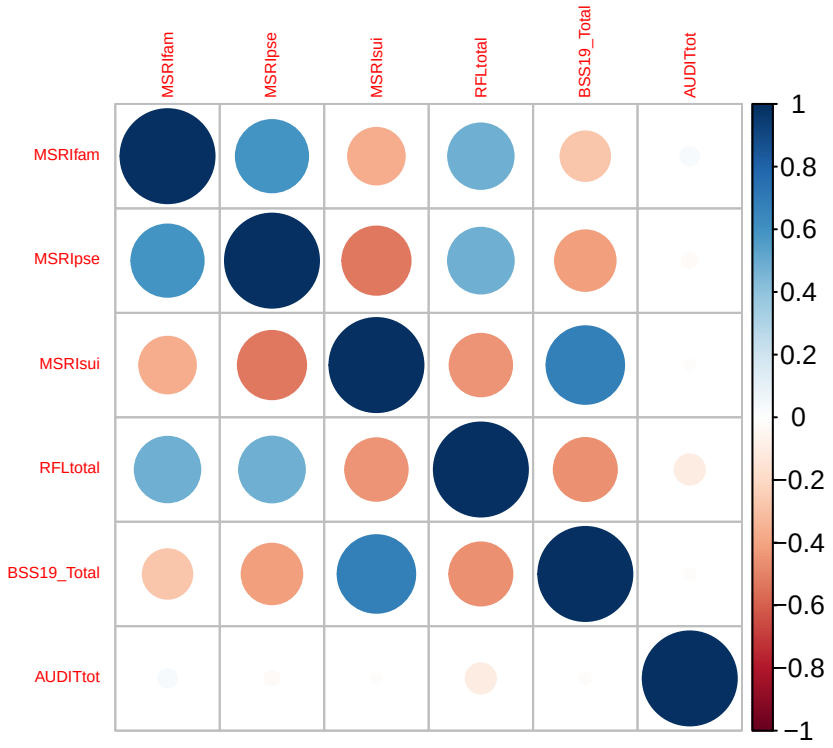
- MSRI-28 Total Suicide Ideation Scores (MSRIsui)
- BSS Total Scores (BSS19_Total)
- AUDIT Total Scores (AUDITtot)

Protective Factors:

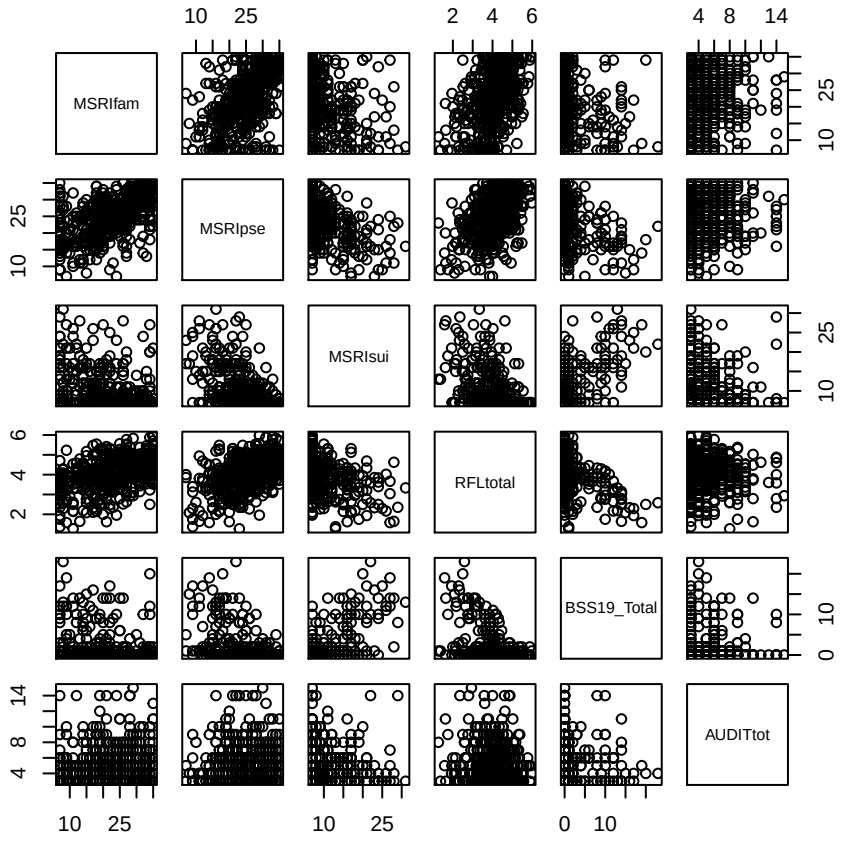
- MSRI-28 Total Family Connectedness Scores (MSRIfam)
- MSRI-28 Total Positive Self-PerceptionScores (MSRIpse)
- RFL Mean Scores (RFLtotal)

Correlation matrix of risk and protective factors' means (regardless of combat group):

```
#get factors of interest
f1 <- MSRI[,c('Study_ID','MSRIfam', 'MSRIpse','MSRIsui')]
f2 <- RFL[,c('Study_ID','RFLtotal')]
f3 <- BSS[,c('Study_ID','BSS19_Total')]
f4 <- Audit[,c('Study_ID','AUDITtot')]
#join factors by Study_ID
pr_f <- inner_join(inner_join(f1, f2, by="Study_ID"),
           inner_join(f3, f4, by="Study_ID"),
           by="Study_ID")

#ensure data is numeric
for(i in colnames(pr_f)){
  pr_f[,i] <- as.numeric(pr_f[,i])
}

#create correlation plots of the variables of interest:

mydata.rcorr = rcorr(as.matrix(pr_f[,-1]))

corrplot::corrplot(mydata.rcorr$r, tl.cex=.5)
```


MSRIfam
MSRIpse
MSRIsui
RFLtotal
BSS19_Total
AUDITtot
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

```
pairs(pr_f[,-1])
```



If two variables are highly correlated, it will likely increase the amount of variation explained by a predictive model, but reduce the interpretability of the model. These issues may need to be explored depending on the types of predictive models being used, the level of collinearity present, and the role of the variables of concern.

- We see what appears to be a positive correlation between mean MSRI Family Connectedness scores and mean MSRI Positive Self-Perception scores

```
print(paste0("r = ",round(cor(MSRI$MSRIfam, MSRI$MSRIpse),4)))
```

```
## [1] "r = 0.5905"
```

- We see what appears to be a positive correlation between mean MSRI Family Connectedness scores and mean RFL scores

```
print(paste0("r = ",round(cor(MSRI$MSRIfam, RFL$RFLtotal),4)))
```

```
## [1] "r = 0.4848"
```

- We see what appears to be a positive correlation between mean MSRI Positive Self-Perception scores and mean RFL scores

```
print(paste0("r = ",round(cor(MSRI$MSRIpse, RFL$RFLtotal),4)))
```

```
## [1] "r = 0.4888"
```

- We see what may be a positive correlation between mean MSRI Suicide Ideation scores and total Beck Scale for Suicide Ideation scores

```r
print(paste0("r = ",round(cor(MSRI$MSRIsui, BSS$BSS19_Total),4)))
```

```
## [1] "r = 0.6807"
```

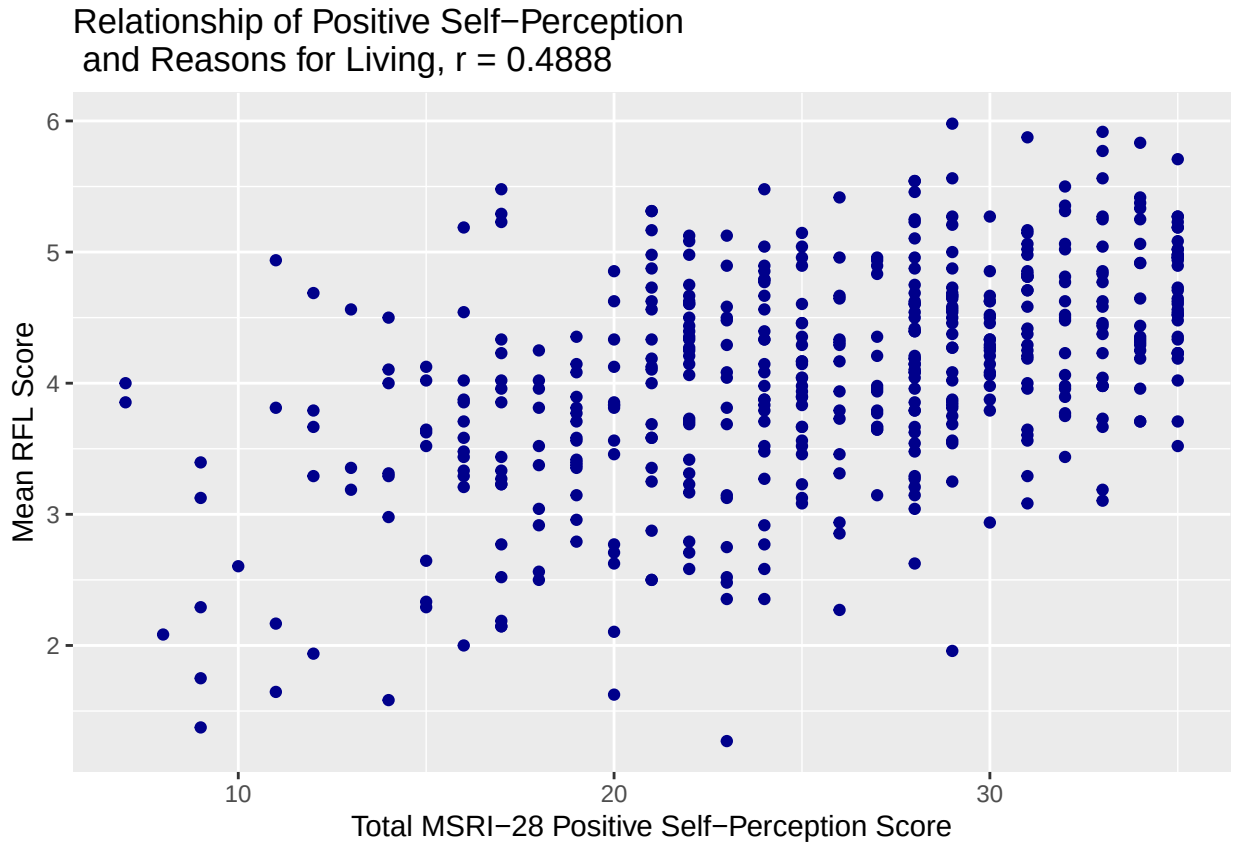
- We see what may be a negative correlation between mean RFL scores and total Beck Scale for Suicide Ideation scores

```r
print(paste0("r = ",round(cor(RFL$RFLtotal, BSS$BSS19_Total),4)))
```

```
## [1] "r = -0.4501"
```

Further exploring correlation of MSRI-28 Positive Self-Perception and the Reasons for Living Inventory (regardless of combat group):

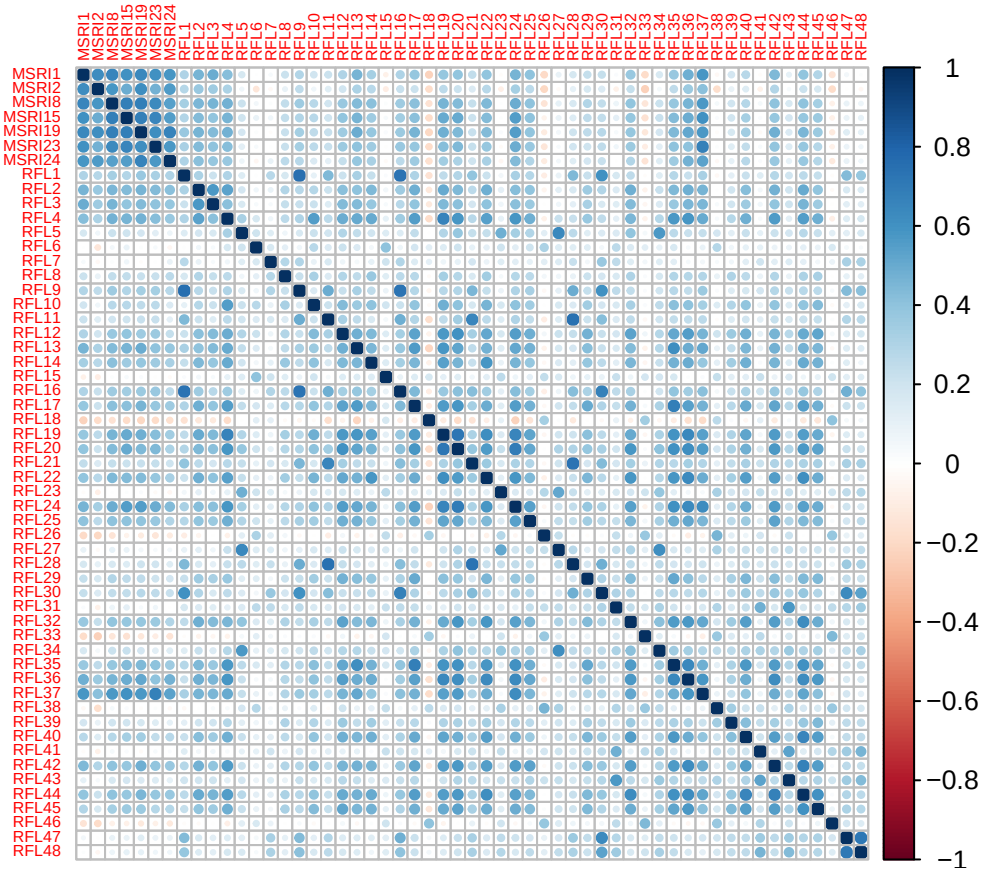
```
ms1 <- MSRI[,c("Study_ID","MSRIpse")]
rf1 <-RFL[,c("Study_ID","RFLtotal")]
ms_rf <- inner_join(ms1, rf1, by="Study_ID")

ggplot(data=ms_rf, aes(x=MSRIpse,y=RFLtotal))+
  geom_point(color="dark blue")+
  xlab("Total MSRI-28 Positive Self-Perception Score")+
  ylab("Mean RFL Score")+
  ggtitle(paste0("Relationship of Positive Self-Perception\n and Reasons for Living",
                 ", r = ", round(cor(ms_rf$MSRIpse, ms_rf$RFLtotal),4)))
```



Further Granularity: Exploring the relationship of MSRI-28 Positive Self-Perception individual question responses and Reasons for Living Inventory individual question responses, regardless of combat group:

```
#Study_ID, MSRI1,MSRI2,MSRI8,MSRI15,MSRI19,MSRI23,MSRI24 from MSRI
MSRI2 <- get_df('MSRI')
MSRI_psp <- MSRI2[,c('Study_ID','MSRI1','MSRI2','MSRI8',
                     'MSRI15','MSRI19','MSRI23','MSRI24')]

#join by Study_ID to RFL1-RFL48
RFL2 <- get_df('RFL')
RFL_reasons <- RFL2[,c("Study_ID",paste0("RFL", as.character(1:48)))]
```

```
cor1 <- inner_join(MSRI_psp, RFL_reasons, by="Study_ID")

for(i in colnames(cor1)){
  cor1[,i] <- as.numeric(cor1[,i])
}

mydata.rcorr = rcorr(as.matrix(cor1[,-1]))

corrplot::corrplot(mydata.rcorr$r, tl.cex=.5)
```



Hypothesis: The correlation between RFL and MSRI Suicide Ideation will be significantly different for veterans who experienced combat from veterans who did not experience combat.

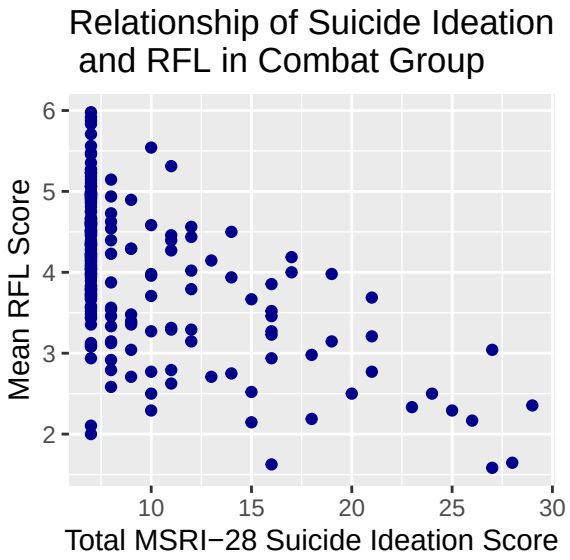
Does experiencing combat affect the correlation between reasons for living and suicide ideation?

Independent Correlation Analysis: comparing the correlation of RFL with MSRI suicide ideation for combat group vs. non-combat group:

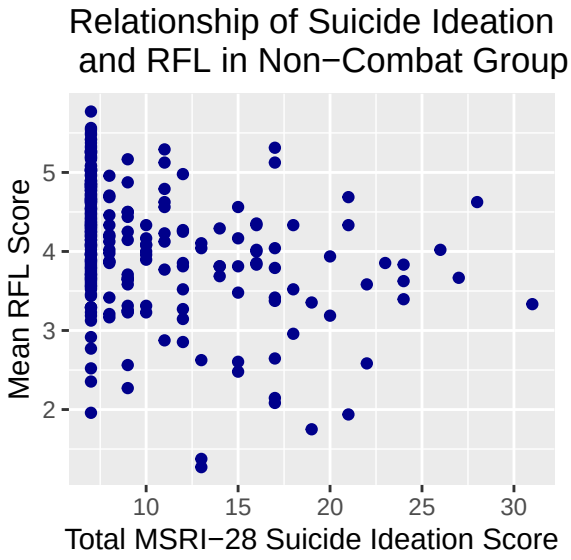
Method: Fisher's r to z transformation

Null hypothesis: correlations are *not* significantly different

```
#subset the variables of interest
ms2 <- MSRI2[,c("Study_ID","MSRIsui")]
rf2 <-RFL[,c("Study_ID","RFLtotal")]
group_id <- osb_gen[,c('Study_ID','Combat')]

#join variables by Study_ID
ms_rf2 <- inner_join(ms2, rf2, by="Study_ID")
ms_rf2 <- left_join(ms_rf2, group_id, by="Study_ID")

#plot data of interest for combat group
ggplot(data=ms_rf2[ms_rf2$Combat==1,],
             aes(x=MSRIsui,y=RFLtotal))+
  geom_point(color="dark blue")+
  xlab("Total MSRI-28 Suicide Ideation Score")+
  ylab("Mean RFL Score")+
  ggtitle("Relationship of Suicide Ideation\n and RFL in Combat Group")
```




```
#plot data of interest for non-combat group
ggplot(data=ms_rf2[ms_rf2$Combat==2,],
             aes(x=MSRIsui,y=RFLtotal))+
  geom_point(color="dark blue")+
  xlab("Total MSRI-28 Suicide Ideation Score")+
  ylab("Mean RFL Score")+
  ggtitle("Relationship of Suicide Ideation\n and RFL in Non-Combat Group")
```




```
#get the correlation for each group
r1 <- cor(x=ms_rf2$MSRIsui[ms_rf2$Combat==1],
              y=ms_rf2$RFLtotal[ms_rf2$Combat==1])
r2 <- cor(x=ms_rf2$MSRIsui[ms_rf2$Combat==2],
              y=ms_rf2$RFLtotal[ms_rf2$Combat==2])
#print the correlation for each group
writeLines(paste0("for combat group: r = ", round(r1,4),
             "\n",
             "for non-combat group: r = ", round(r2,4)))
```

```
## for combat group: r = -0.5658
## for non-combat group: r = -0.3349
```

```
#create a function to transform the correlation score for each group
transform_z <- function(r){
  z=.5*(log(1+r)-log(1-r))
  return(z)
}
#transform each correlation to a z-score using transform_z
z1 <- transform_z(r1)
z2 <- transform_z(r2)
#print the z-score for each group
writeLines(paste0("combat group: z = ", round(z1,4),
             "\n",
             "non-combat group: z = ", round(z2,4)))
```

```
## combat group: z = -0.6414
## non-combat group: z = -0.3484
```

```
#get the number of observations in each group
n2 <- length(ms_rf2$MSRIsui[ms_rf2$Combat==2]) # without combat
n1 <- length(ms_rf2$MSRIsui[ms_rf2$Combat==1]) # with combat
#get the observed z score
z_obs <- (z1-z2) / sqrt((1/(n1-3))+(1/(n2-3)))
#print the observed z score
print(paste0("Test Statistic: ", z_obs))
```

```
## [1] "Test Statistic: -3.15484676037578"
```

Compare the Test Statistic Z against the critical value. Assuming alpha = .05, the critical value is positive or negative 1.96. Is the Test Statistic in the rejection region?

```
#for significance of .05, critical z val is +/- 1.96
#null hypothesis: correlations aren't significantly different
print(abs(z_obs)>abs(1.96))
```

```
## [1] TRUE
```

```
#rejection region - reject the null
#infer that correlations are significantly different two groups
```

Therefore we reject the null and infer that correlations are significantly different between the two groups.

Hypothesis: Veterans who have not been exposed to combat will be more likely to have a higher mean score on the MSRI-28 Family Connectedness scale than those who have been exposed to combat.

```
#data manipulation: get variables of interest and order by ID
gen_types <- osb_gen %>%
  dplyr::select(Study_ID, Combat)%>%
  dplyr::arrange(Study_ID)


#join MSRIfam with Combat by Study_ID
msri_fam <- MSRI[,c('Study_ID','MSRIfam')]
fam_combat <- merge(msri_fam, gen_types, by="Study_ID")

#data manipulation: ensure variables are coded as factors
fam_combat$Combat <- as.factor(fam_combat$Combat)
fam_combat$Combat <- ordered(fam_combat$Combat,
                             levels = c(1,2))

#get the averages rather than the totals by dividing by 7
#There are 7 MSRI Family Connectedness questions
group1_msri <- fam_combat$MSRIfam[fam_combat$Combat=="1"]/7
group2_msri <- fam_combat$MSRIfam[fam_combat$Combat=="2"]/7

#get summary statistics of the new data
sumstats <- dplyr::group_by(fam_combat, Combat) %>%
  dplyr::summarise(
    count = n(),
    mean = mean(MSRIfam, na.rm = TRUE),
    sd = sd(MSRIfam, na.rm = TRUE)
  )

#data manipulation: make the summary statistics a data frame
sumstats <- as.data.frame(sumstats)

#make the data frame into a nice-looking table to output
kable(sumstats, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

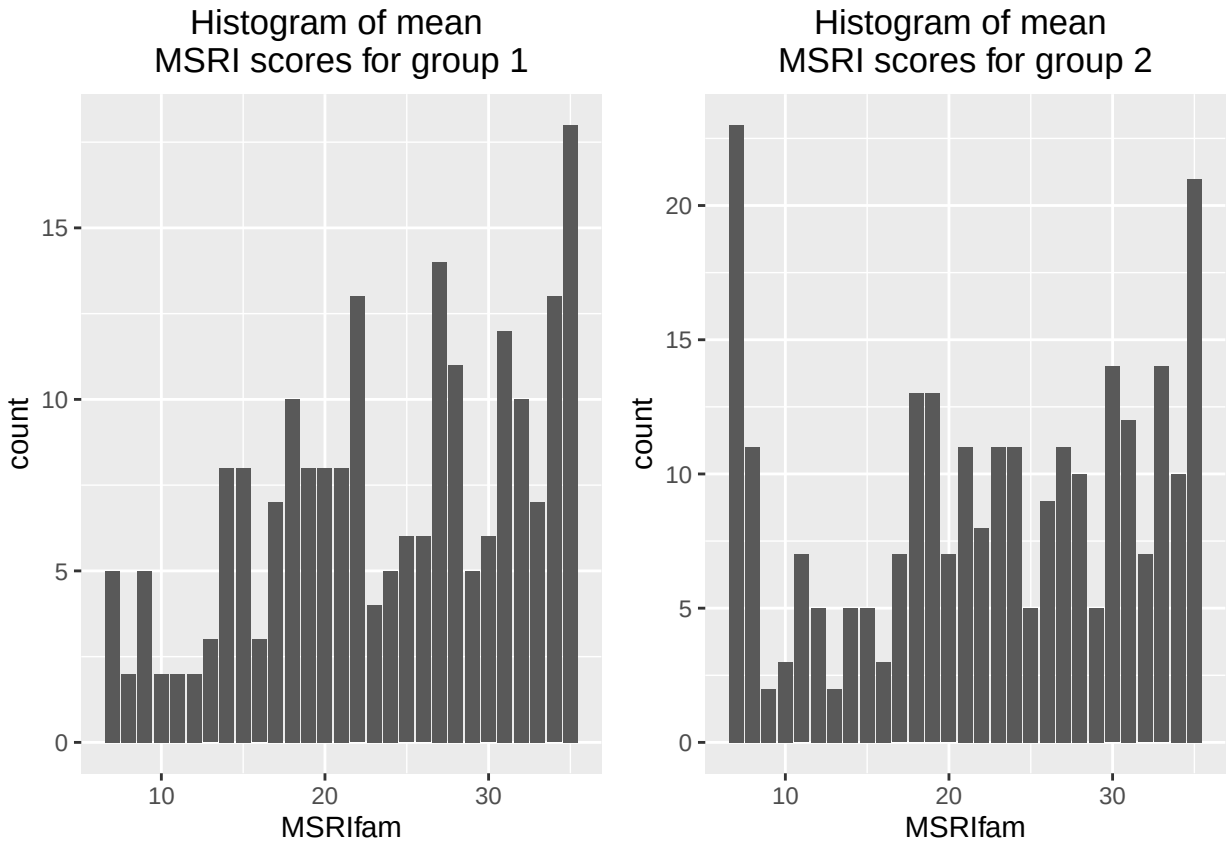
Combat	count	mean	sd
1	211	24.02844	7.930601
2	265	22.45660	8.948877

```
#output: summary statistics table (mean, count, sd)
```

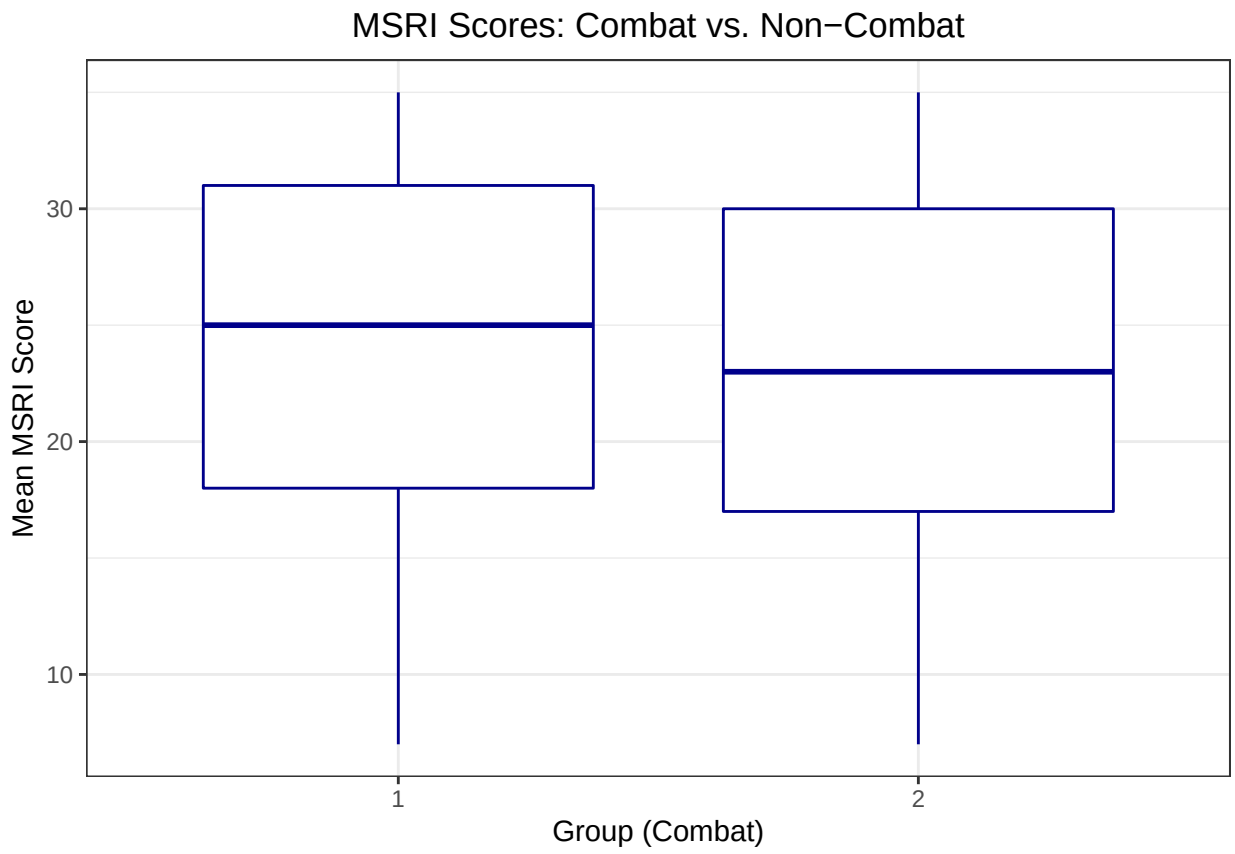
Looking for normality:

```
#plot histogram for group 1
p1 <- ggplot(fam_combat[fam_combat$Combat==1,], aes(x=MSRIfam))+
  geom_histogram(stat="count")+
  ggtitle("Histogram of mean \nMSRI scores for group 1")+
```

```
  theme(plot.title = element_text(hjust = 0.5))
#plot histogram for group 2
p2 <-ggplot(fam_combat[fam_combat$Combat==2,], aes(x=MSRIfam))+
  geom_histogram(stat="count")+
  ggtitle("Histogram of mean \nMSRI scores for group 2")+
  theme(plot.title = element_text(hjust = 0.5))
#print histograms side by side
grid.arrange(p1, p2, ncol = 2)
```




```
#plot Combat vs. Non-Combat MSRI Scores
ggplot(fam_combat, aes(x=Combat, y=MSRIfam))+
  geom_boxplot(color="darkblue")+
  xlab("Group (Combat)")+
  ylab("Mean MSRI Score")+
  theme_bw()+
  ggtitle("MSRI Scores: Combat vs. Non-Combat")+
  theme(plot.title = element_text(hjust = 0.5))
```



T-test for mean MSRI Scores between veterans who have been exposed to combat and veterans who have not been exposed to combat:

```
t.test(group1_msri, group2_msri)
```

```
##
##  Welch Two Sample t-test
##
## data:  group1_msri and group2_msri
## t = 2.0288, df = 468.56, p-value = 0.04305
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.007052787 0.442042140
## sample estimates:
## mean of x mean of y
##  3.432634  3.208086
```

Conclusion:

Considering a p-value of .05, we reject the null hypothesis that $\mu_1 = \mu_2$ and conclude that the mean MSRI Score is significantly different for those who experienced combat from those who did not experience combat.

Question: **How certain are we? Consider the p-value resulting from the t-test.**

Question: **Was the t-test's assumption of normality met for this data? How important is this assumption? What is a non-parametric alternative?**

- Alternative: Mann-Whitney-Wilcoxon Test:

```
wilcox.test(group1_msri, group2_msri)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  group1_msri and group2_msri
## W = 30486, p-value = 0.08956
## alternative hypothesis: true location shift is not equal to 0
```

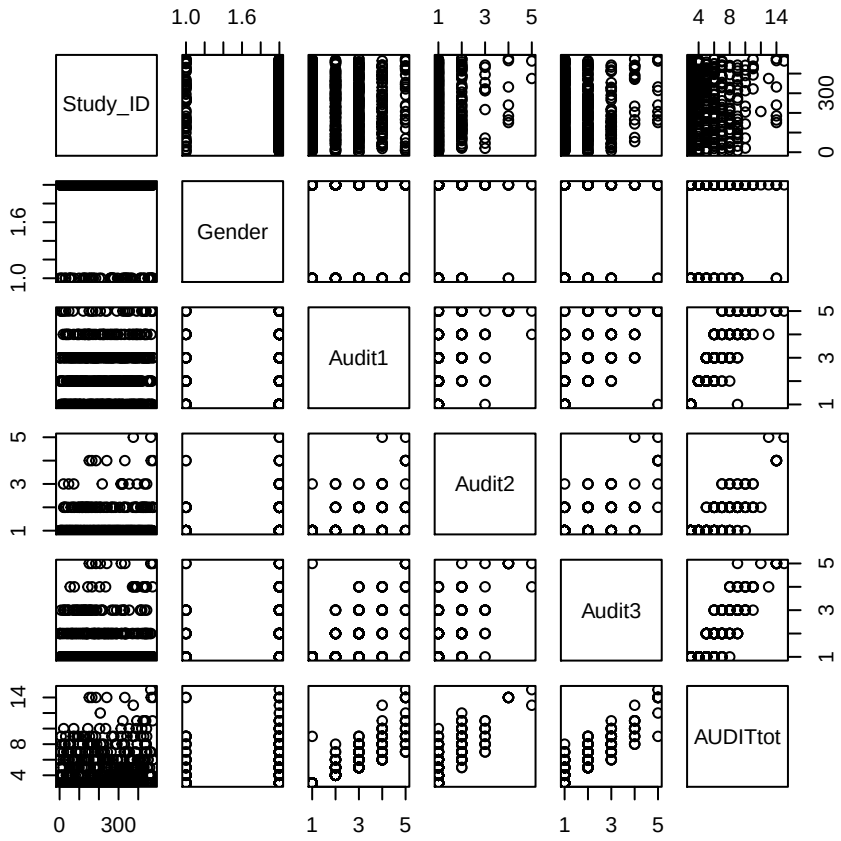
Conclusion from a Mann-Whitney-Wilcoxon Test:

We do not reject the null hypothesis that $\mu_1 = \mu_2$ and conclude that the mean MSRI Family Connectedness Score is not significantly different for those who experienced combat from those who did not experience combat.

Hypothesis: Veterans who have been exposed to combat will be more likely to have a higher mean score for a problematic drinking scale than those who have not been exposed to combat.

Alcohol Use Disorders Test: self-report measure of problematic drinking for adults:

```
#get pair plot of Audit data
pairs(Audit)
```

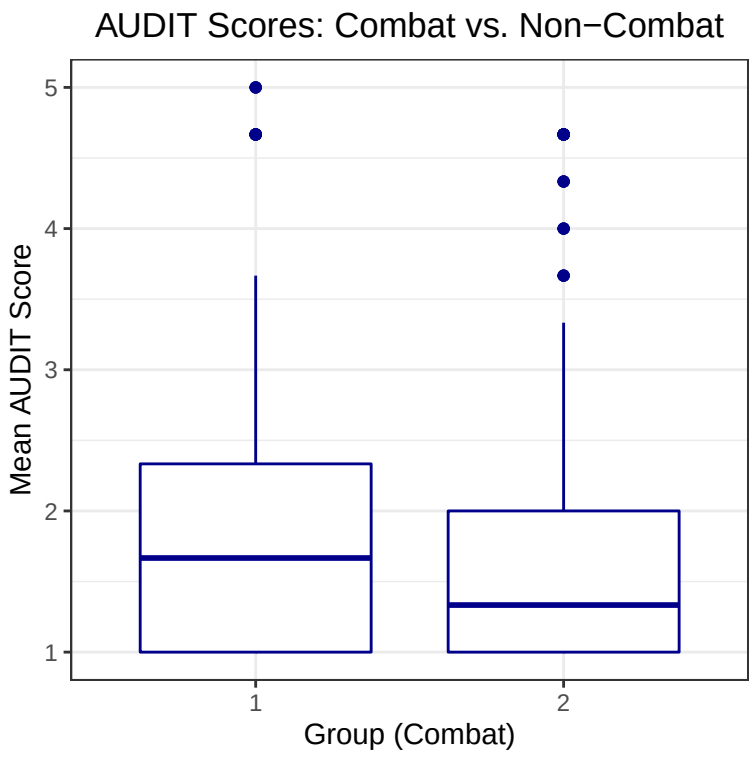



```
#data manipulation: melt the data
audit_melt <- reshape2::melt(Audit[,c("Study_ID","Audit1", "Audit2", "Audit3")],
                             id.vars=c("Study_ID"))
#data manipulation: group the data by Study_ID and add the mean score for each
Audit2 <- audit_melt %>%
  dplyr::group_by(Study_ID) %>%
  dplyr::mutate(aud_mean = mean(value))

#data manipulation: merge (join) the Audit data with combat / general data by Study_ID
Audit2 <- merge(Audit2, gen_types, by= "Study_ID")
Audit2$group <- ordered(Audit2$Combat,
                        levels = c(1,2))
#get means for each group
group1_Audit <- Audit2$aud_mean[Audit2$group=="1"]
group2_Audit <- Audit2$aud_mean[Audit2$group=="2"]
```

```r
#summarize Audit and group data
sumstats <- dplyr::group_by(Audit2, group) %>%
  dplyr::summarise(
    count = n(),
    mean = mean(aud_mean, na.rm = TRUE),
    sd = sd(aud_mean, na.rm = TRUE)
  )

#create a table of summary statistics
sumstats <- as.data.frame(sumstats)
kable(sumstats, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

group	count	mean	sd
1	633	1.805687	0.8006124
2	795	1.645283	0.7871527

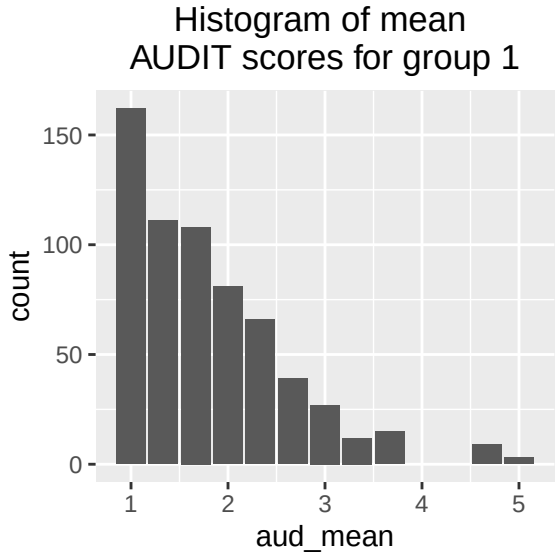
```r
#plot Combat vs. Non-Combat AUDIT Scores
ggplot(Audit2, aes(x=group, y=aud_mean))+
  geom_boxplot(color="darkblue")+
  xlab("Group (Combat)")+
  ylab("Mean AUDIT Score")+
  theme_bw()+
  ggtitle("AUDIT Scores: Combat vs. Non-Combat")+
  theme(plot.title = element_text(hjust = 0.5))
```



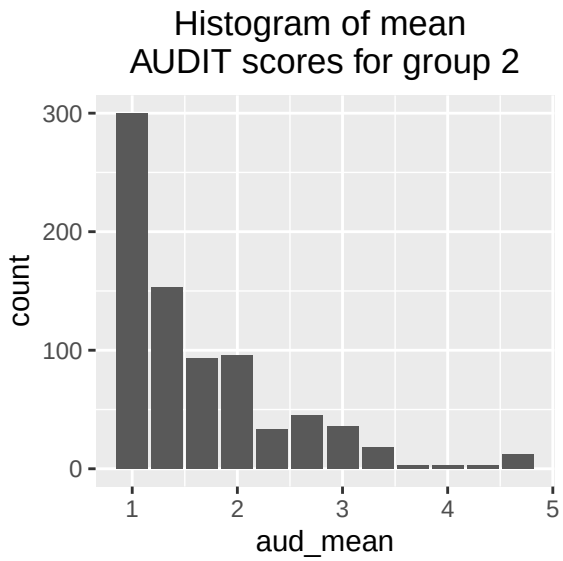
```
#output: summary stats and boxplots
```

Looking for normality:

```
#plot histogram for group 1
ggplot(Audit2[Audit2$Combat==1,], aes(x=aud_mean))+
  geom_histogram(stat="count")+
  ggtitle("Histogram of mean \nAUDIT scores for group 1")+
  theme(plot.title = element_text(hjust = 0.5))
```




```
#plot histogram for group 2
ggplot(Audit2[Audit2$Combat==2,], aes(x=aud_mean))+
  geom_histogram(stat="count")+
  ggtitle("Histogram of mean \nAUDIT scores for group 2")+
  theme(plot.title = element_text(hjust = 0.5))
```



T-test for mean AUDIT Scores between veterans who have been exposed to combat and veterans who have not been exposed to combat:

```
t.test(group1_Audit, group2_Audit)
```

```
##
##  Welch Two Sample t-test
##
## data:  group1_Audit and group2_Audit
## t = 3.7892, df = 1345, p-value = 0.0001578
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.07736046 0.24344791
## sample estimates:
## mean of x mean of y
##  1.805687  1.645283
```

Conclusion from a t-test:

We reject the null hypothesis that $\mu_1 = \mu_2$ and conclude that the mean AUDIT Score is significantly different for those who experienced combat from those who did not experience combat.

Question: **Was the t-test's assumption of normality met for this data? How important is this assumption? What is a non-parametric alternative?**

- Alternative: Mann-Whitney-Wilcoxon Test:

```
wilcox.test(Audit2$aud_mean ~ Audit2$Combat)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Audit2$aud_mean by Audit2$Combat
## W = 289040, p-value = 7.5e-07
## alternative hypothesis: true location shift is not equal to 0
```

Conclusion from a Mann-Whitney-Wilcoxon Test:

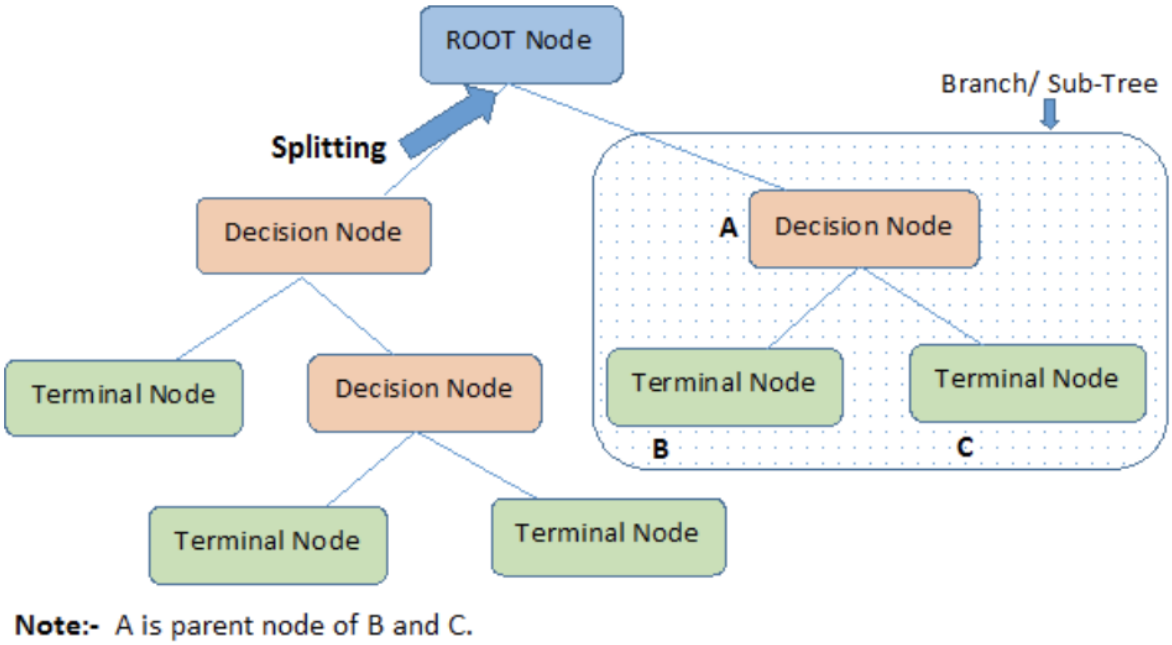
We reject the null hypothesis that $\mu_1 = \mu_2$ and conclude that the mean AUDIT Score is significantly different for those who experienced combat from those who did not experience combat.

Predicting Suicide Attempts with Decision Trees:

Motivation: determining how mean MSRI-28 scores impact the likelihood of a veteran attempting suicide

What other relationships might exist?

Concept: Classify a categorical response based from predictor variables using decision trees. The structure of a decision tree can be seen in this example from Analytics Vidhya:



* Example from Medium: Analytics Vidhya

How is the tree constructed? Decision tree algorithms can use different types of decision-making criteria to determine how to best split information for classification.

Decision-Making Criteria: Entropy and Information Gain

In this analysis, the decisions will be based on *information gain*, which is a function of *entropy*.

The *Entropy* of a set of information is defined as

$$Entropy = -\sum_{i=1}^{c} P(V_i) \times log_2(P(V_i))$$

in which there are c unique values in a set.

Example of entropy:

For example, in a set $\{yes, yes, no, yes, no, no\}$, there are 3 "yes" values and 3 "no" values. The probability of drawing either is 3/6. Let's see what entropy looks like using the formula above:

```
-1*((3/6)*log2(3/6)+(3/6)*log2(3/6))
```

```
## [1] 1
```

We find that this results in maximum entropy, or uncertainty, about the data set.

The **information gain** is the *expected reduction in entropy caused by partitioning the examples according to a given attribute (F. Aiolli, Dipartimento di Matematica Pura ed Applicata Università di Padova).* It is equal to the entropy of a parent node minus the weighted entropy of the children nodes. In each split, we seek to maximize the information gain. This is how decisions in the following decision trees will be made.

Example of information gain:

Let's evaluate the following data to understand information gain:

```
#generate example table from https://www.math.unipd.it/~aiolli/corsi/0708/IR/Lez12.pdf
example_data<- data.frame(attribute1=c("A","B"),positive=c(6,3),negative=c(2,5))
#make the summary data frame into a nice-looking table to output
kable(example_data, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(font_size = 8)
```

attribute1	positive	negative
A	6	2
B	3	5

In the above table:

- There are 8 instances of "A" and 8 instances of "B". Thus, P(A)=8/16, or .5, and P(B)=8/16, or .5.
- There are 9 positive results and 7 negative results.

To obtain the entropy of the set, we use the formula above.

```
entropy_of_set <- -1*((9/16)*log2(9/16)+(7/16)*log2(7/16))
print(entropy_of_set)
```

```
## [1] 0.9886994
```

To get the entropy of attribute1, we will sum the weighted entropy of each factor in that attribute. To do this, we must first obtain the entropy of A and of B.

To obtain the entropy of A, we use the formula above applied to the "A" data subset. As mentioned, there are 8 instances of "A", where 2 of them had negative results and 6 had positive results. Thus, the entropy of "A" can be calculated as:

```
entropy_of_A <- -1*((2/8)*log2(2/8)+(6/8)*log2(6/8))
print(entropy_of_A)
```

```
## [1] 0.8112781
```

Similarly, the entropy of "B" can be calculated as:

```
entropy_of_B <- -1*((3/8)*log2(3/8)+(5/8)*log2(5/8))
print(entropy_of_B)
```

```
## [1] 0.954434
```

Because 8 of the 16 observations have an attribute1 of "A", the weight for the entropy of A is 8/16. Similarly, because 8 of the 16 observations have an attribute1 of "B", the weight for the entropy of B is 8/16.

```
weight_A <- 8/16
weight_B <- 8/16
```

We can now calculate the entropy of the attribute (or the weighted entropy of the children nodes).

```
entropy_of_attribute1<- weight_A*entropy_of_A+weight_B*entropy_of_B
entropy_of_attribute1
```

```
## [1] 0.8828561
```

Given the entropy of the attribute and the set, we can now calculate information gain for splitting at attribute1 as the entropy of the parent node (in this case, the set) minus the entropy of attribute1.

```
information_gain1 <- entropy_of_set-entropy_of_attribute1
information_gain1
```

```
## [1] 0.1058433
```

This gives us a sense of how much information was gained by choosing to split this data set by attribute1 as the first branch of the decision tree.

Example values derived from F. Aiolli, Dipartimento di Matematica Pura ed Applicata Università di Padova

Choosing Decision-Making Criteria: Information Gain vs. Gini Index

In this analysis, information gain was used to create decision trees which rely on entropy as the splitting criteria.

Other options, such as the Gini Split algorithm, can be used for the same purpose. Unlike information gain, the Gini Split algorithm relies on the Gini Index.

$$Gini = 1 - \sum_{i=1}^{c}(p_i)^2$$

The results from each may differ, though one is not always better than the other. According to Medium, "it isn't a bad idea to use both because the tree classification may vary."

According to learnbymarketing, the Gini Split *favors larger partitions*, while Information gain *favors partitions that have small counts but many distinct values*. To learn more about choosing decision criteria, visit learnbymarketing

Predicting suicide attempts from MSRI and combat exposure

Use a decision tree to classify each observation as either "suicide attempt" or "no attempt" from predictors in MSRI-28:

- suicide ideation mean score (MSRI)
- family connectedness mean score (MSRI)
- negative affect mean score (MSRI)
- positive self-perception mean score (MSRI)
- if veteran has been exposed to combat

Data Pre-Processing

A large part of the data science process involves cleaning data, performing exploratory data analysis, transforming variables, and manipulating data structures.

Look at the summary of the data below. Do you see any potential issues with the variable "suicide attempt"?

```
combat_var <- osb_gen[,c('Study_ID','Combat','Suicide_attempt')]
sum_msri <- MSRI[,c('Study_ID','MSRIsui','MSRIfam','MSRIna','MSRIpse')]
#merge (join) variables of interest from general data frame (Combat, Suicide_attempt)
#to the MSRI scores data frame
msri_combat <- merge(sum_msri, combat_var, by = 'Study_ID', all.x=TRUE, all.y = FALSE)
#make summary table
sumtable <- summary(msri_combat)
#make the summary data frame into a nice-looking table to output
kable(sumtable, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(font_size = 8)
```

Study_ID	MSRIsui	MSRIfam	MSRIna	MSRIpse	Combat	Suicide_attempt
Min. : 1.0	Min. : 7.000	Min. : 7.00	Min. : 7.00	Min. : 7.00	Min. :1.000	Min. : 1.000
1st Qu.:119.8	1st Qu.: 7.000	1st Qu.:17.00	1st Qu.:12.00	1st Qu.:21.00	1st Qu.:1.000	1st Qu.: 1.000
Median :238.5	Median : 7.000	Median :24.00	Median :17.00	Median :26.00	Median :2.000	Median : 2.000
Mean :238.5	Mean : 9.592	Mean :23.15	Mean :17.76	Mean :25.43	Mean :1.557	Mean : 2.916
3rd Qu.:357.2	3rd Qu.:10.000	3rd Qu.:31.00	3rd Qu.:23.00	3rd Qu.:31.00	3rd Qu.:2.000	3rd Qu.: 2.000
Max. :476.0	Max. :31.000	Max. :35.00	Max. :35.00	Max. :35.00	Max. :2.000	Max. :99.000
NA	NA	NA	NA	NA	NA	NA's :2

One issue with the feature "Suicide Attempt" data set is missingness (NAs) or unexplained values, such as the maximum value seen here 99. This occurs frequently in data science, and if a large amount of data is missing, should be handled by looking for systematic relationships in the data to determine the type of missingness (e.g. missing completely at random, missing at random, or missing not at random) before moving on. Certain cases can be addressed through replacing missing data with a median value or the result from a simple linear regression, while others may require multiple and multivariate imputation methods.

In this case, there are very few (less than 5%) missing or unexplained suicide attempt values, so listwise deletion will be used. This means **we will completely remove any observation from the data set where Suicide_attempt is *NA* or *99*.**

Feature Selection

Another consideration in this data set is which variables should be used as predictors. For instance, predictors could be each individual question from the MSRI, total MSRI scores for each type of factor, or mean MSRI scores for each type of factor.

In this case we will explore using the mean and total MSRI scores rather than pulling each individual

question's response. This is because we prefer to rely on higher-level data (e.g. total or mean) to improve model explainability. This is a good way to begin, though if our model does not do a good job explaining our data, we may need to use individual scores for more granularity.

Note that the MSRI scores are divided by 7 (as there are 7 MSRI questions) below to obtain the mean score rather than the total score.

```
#make a copy of msri_combat without data transformation for later use
msri_combat_copy <- msri_combat

#want to get the averages rather than the totals by dividing by 7
msri_combat$MSRIsui <- msri_combat$MSRIsui/7
msri_combat$MSRIfam <- msri_combat$MSRIfam/7
msri_combat$MSRIna <- msri_combat$MSRIna/7
msri_combat$MSRIpse <- msri_combat$MSRIpse/7

#removing data that is missing
#this is called "complete case analysis"
#it is acceptable because we are missing very little data
#and we have a good sample size
msri_combat <- msri_combat[msri_combat$Suicide_attempt!="99",]
msri_combat <- msri_combat[-which(is.na(msri_combat$Suicide_attempt)),]

#recode the variables to be easy to read and understand
#they can be strings or numbers as long as they are used as factors
msri_combat$Suicide_attempt[msri_combat$Suicide_attempt==2] <- "no attempt"
msri_combat$Suicide_attempt[msri_combat$Suicide_attempt==1] <- "attempt"

#renaming the variables in msri_combat (e.g. capitalizing "A" in Suicide_attempt)
names(msri_combat) <- c('Study_ID','Sui_Ideation', 'Family_Connectedness',
                        'Neg_Affect','Pos_Self_Perception', 'Combat',
                        'Suicide_Attempt')

#the data to be used for predicting suicide attempt
msri_head <- head(msri_combat)
#view msri_head in a nice-looking table output
kable(msri_head, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(font_size = 8)
```

Study_ID	Sui_Ideation	Family_Connectedness	Neg_Affect	Pos_Self_Perception	Combat	Suicide_Attempt
1	1.000000	3.857143	2.285714	3.428571	2	attempt
2	1.000000	3.571429	3.714286	4.000000	2	no attempt
3	1.714286	1.142857	2.714286	3.142857	2	attempt
4	1.000000	2.857143	2.285714	4.571429	2	no attempt
5	1.000000	1.000000	1.857143	3.285714	2	no attempt
6	2.285714	2.000000	2.714286	2.428571	1	no attempt

Predicting suicide attempts from MSRI mean and combat exposure:

```
#set seed so that results are reproducible
set.seed(123)
#split the data into testing and training sets (70/30 split)
split = sample.split(msri_combat$Suicide_Attempt, SplitRatio = 0.7)
#get training data
train = subset(msri_combat, split==TRUE)
#get testing data
test = subset(msri_combat, split==FALSE)
#for prediction:
#remove Study_ID as it is not a predictor
#remove Suicide_Attempt since that's the response we want to predict
remove_columns <- which(names(msri_combat) %in% c('Study_ID','Suicide_Attempt'))


#get decision tree
#using information gain
tree_ig = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.02),
             parms = list(split = 'information'))
#using gini index
tree_gini = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.02),
             parms = list(split = 'gini'))


#predict the results of the test set using each type of tree (gini, information gain)
#information gain
tree_ig.pred = predict(tree_ig, newdata=test[,-remove_columns], type='class')
#gini
tree_gini.pred = predict(tree_gini, newdata=test[,-remove_columns], type='class')
```

Evaluating the prediction results

```
#get a confusion matrix to view the results of the information gain prediction
x <- confusionMatrix(as.factor(test$Suicide_Attempt), as.factor(tree_ig.pred))
#store the results as a data frame to easily access variables
#note: reference refers to ground truth
pred_results<- data.frame(x$table)

#get values for precision and recall
TP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="attempt",
             'Freq']
FP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
TN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
FN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="attempt",
             'Freq']
```

```
#print the prediction accuracy: (TP+TN) / (TP + FP + TN + FN)
accuracy <- (TP+TN) / (TP + FP + TN + FN)
paste0("prediction accuracy: ",round(accuracy,digits = 2))
```

```
## [1] "prediction accuracy: 0.76"
```

```
#Precision: TP / (TP + FP)
precision <- TP/(TP+FP)
paste0("precision: ",round(precision,digits = 2))
```

```
## [1] "precision: 0.34"
```

```
#Recall: TP / (TP + FN)
recall <- TP/(TP+FN)
paste0("recall: ",round(recall,digits = 2))
```

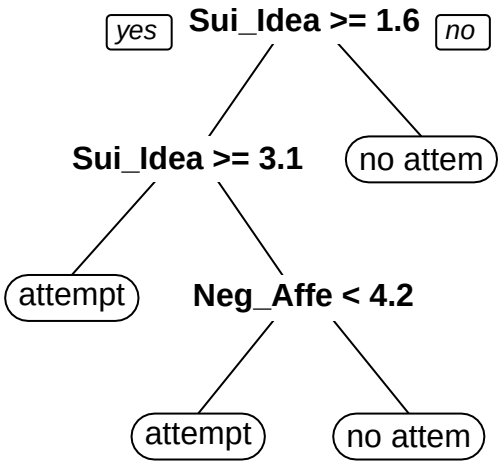
```
## [1] "recall: 0.79"
```

```
#print confusion matrix
kable(x$table, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	attempt	no attempt
attempt	15	29
no attempt	4	92

How is the decision tree classifying these participants?

```
#show decision tree (based on information gain)
prp(tree_ig)
```



If mean suicide ideation score is less than 1.6, our decision tree model would predict that there will be no suicide attempt. If the mean suicide ideation score is 3.1 or greater, our decision tree model would predict that there will be a suicide attempt. If the mean suicide ideation score is between 1.6 and 3.1, the prediction of a suicide attempt depends on mean MSRI-Negative Affect score, with a mean negative affect score of 4.1 or greater leading to the prediction of a suicide attempt.

What is gained from this?

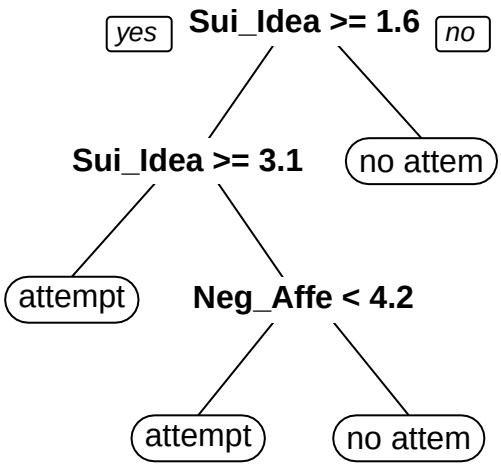
- We can understand how our model makes decisions and which variables are important predictors. To take a closer look at the relative importance of each variable as a predictor of suicide attempt, we can investigate the variable importance based on the information gain each factor provides:

```
a <- tree_ig$variable.importance
a <- as.data.frame(a)
names(a) <- c("variable importance")

kable(a, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	variable importance
Sui_Ideation	17.250597
Neg_Affect	4.823566
Pos_Self_Perception	3.172527
Family_Connectedness	1.057509

What if, instead of information gain, we used a different decision criteria such as the gini index to determine splits?

```
#show decision tree (based on gini index)
prp(tree_gini)
```



Data Transformation: *total* vs. *mean* score results

Will the results change if we used the total MSRI scores rather than the mean scores?

Use a decision tree to classify each observation as either "suicide attempt" or "no attempt" from predictors in MSRI-28:

- suicide ideation total score (MSRI)
- family connectedness total score (MSRI)
- negative affect total score (MSRI)
- positive self-perception total score (MSRI)
- if veteran has been exposed to combat

Predicting suicide attempts from MSRI total and combat exposure:

```
#return msri_combat to original state to obtain totals rather than means
msri_combat <- msri_combat_copy

#re-clean the data since msri_combat_copy was used
msri_combat <- msri_combat[msri_combat$Suicide_attempt!="99",]
msri_combat <- msri_combat[-which(is.na(msri_combat$Suicide_attempt)),]

msri_combat$Suicide_attempt[msri_combat$Suicide_attempt==2] <- "no attempt"
msri_combat$Suicide_attempt[msri_combat$Suicide_attempt==1] <- "attempt"

names(msri_combat) <- c('Study_ID','Sui_Ideation', 'Family_Connectedness',
                        'Neg Affect','Pos Self-Perception', 'Combat',
                        'Suicide_Attempt')

#the data to be used for predicting suicide attempt
msri_head <- head(msri_combat)
#view msri_head in a nice-looking table output
kable(msri_head, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(font_size = 8)
```

Study_ID	Sui_Ideation	Family_Connectedness	Neg Affect	Pos Self-Perception	Combat	Suicide_Attempt
1	7	27	16	24	2	attempt
2	7	25	26	28	2	no attempt
3	12	8	19	22	2	attempt
4	7	20	16	32	2	no attempt
5	7	7	13	23	2	no attempt
6	16	14	19	17	1	no attempt

```
#set seed so that results are reproducible
set.seed(123)
#split the data into testing and training sets (70/30 split)
split = sample.split(msri_combat$Suicide_Attempt, SplitRatio = 0.7)
#get training data
train = subset(msri_combat, split==TRUE)
#get testing data
test = subset(msri_combat, split==FALSE)
#for prediction:
#remove Study_ID as it is not a predictor
#remove Suicide_Attempt since that's the response we want to predict
```

```
remove_columns <- which(names(msri_combat) %in% c('Study_ID','Suicide_Attempt'))


#get decision tree
#using gini index
tree_ig = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.02),
             parms = list(split = 'information'))
#using information gain
tree_gini = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.02),
             parms = list(split = 'gini'))

#predict the results of the test set using each type of tree (gini, information gain)
#information gain
tree_ig.pred = predict(tree_ig, newdata=test[,-remove_columns], type='class')
#gini
tree_gini.pred = predict(tree_gini, newdata=test[,-remove_columns], type='class')
```

Evaluating the prediction results

```
#get a confusion matrix to view the results of the information gain prediction
x <- confusionMatrix(as.factor(test$Suicide_Attempt), as.factor(tree_ig.pred))
#store the results as a data frame to easily access variables
#note: reference refers to ground truth
pred_results<- data.frame(x$table)

#get values for precision and recall
TP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="attempt",
             'Freq']
FP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
TN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
FN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="attempt",
             'Freq']

#print the prediction accuracy: (TP+TN) / (TP + FP + TN + FN)
accuracy <- (TP+TN) / (TP + FP + TN + FN)
paste0("prediction accuracy: ",round(accuracy,digits = 2))
```

```
## [1] "prediction accuracy: 0.76"
```

```
#Precision: TP / (TP + FP)
precision <- TP/(TP+FP)
paste0("precision: ",round(precision,digits = 2))
```

```
## [1] "precision: 0.34"
```

```
#Recall: TP / (TP + FN)
recall <- TP/(TP+FN)
```

```
paste0("recall: ",round(recall,digits = 2))
```

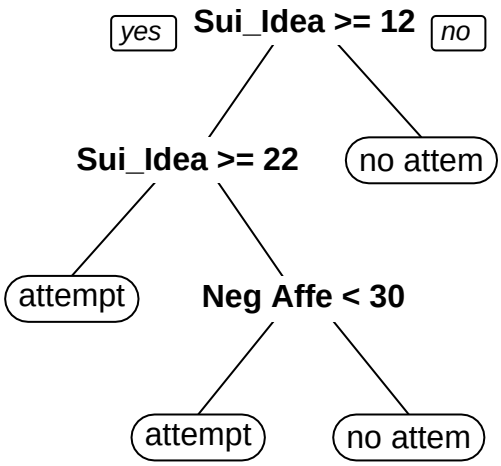
```
## [1] "recall: 0.79"
```

```
#print confusion matrix
kable(x$table, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	attempt	no attempt
attempt	15	29
no attempt	4	92

How is the decision tree classifying these participants?

```
prp(tree_ig)
```



If total suicide ideation score is less than 12, our decision tree model would predict that there will be no suicide attempt. If the total suicide ideation score is 22 or greater, our decision tree model would predict that there will be a suicide attempt. If the total suicide ideation score is between 12 and 22, the prediction of a suicide attempt depends on total MSRI-Negative Affect score, with a total negative affect score of 29 or greater leading to the prediction of a suicide attempt.

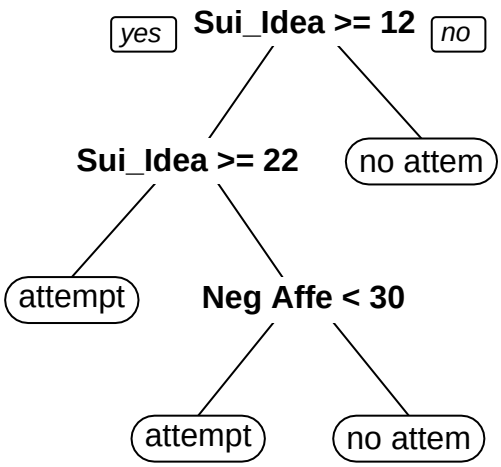
Variable Importance:

```
a <- tree_ig$variable.importance
a <- as.data.frame(a)
names(a) <- c("variable importance")

kable(a, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

What if, instead of information gain, we used the gini index to determine splits?

	variable importance
Sui_Ideation	17.250597
Neg Affect	4.823566
Pos Self-Perception	3.172527
Family_Connectedness	1.057509

```
prp(tree_gini)
```




```
a <- tree_gini$variable.importance
a <- as.data.frame(a)
names(a) <- c("variable importance")

kable(a, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	variable importance
Sui_Ideation	15.6220566
Neg Affect	4.5058647
Pos Self-Perception	2.8985735
Family_Connectedness	0.9661912

- *Note the change in variable importance based on different decision criteria.*

Predicting suicide attempts from measures related to protective factors and risk factors:

Use decision trees to classify each observation as either "suicide attempt" or "no attempt" from measures related to protective factors and risk factors:

- family connectedness mean score (MSRI)
- positive self-perception mean score (MSRI)
- suicide ideation mean score (MSRI)
- reasons for living total score (RFL)
- suicide ideation total score (BSS)
- alcohol use disorders total score (AUDIT)
- if veteran has been exposed to combat

Data Pre-Processing

```
#gather variables of interest
combat_var <- osb_gen[,c('Study_ID','Combat','Suicide_attempt')]
rfl_vars <- RFL[,c('Study_ID','RFLtotal')]
sum_msri <- MSRI[,c('Study_ID','MSRIfam','MSRIpse','MSRIsui')]
bss_vars <- BSS[,c('Study_ID','BSS19_Total')]
audit_vars <- Audit[,c('Study_ID','AUDITtot')]
#join data frames togother by Study_ID
msri_combat <- merge(sum_msri, combat_var, by = 'Study_ID', all.x=TRUE, all.y = FALSE)
msri_combat <- merge(msri_combat, rfl_vars, by = 'Study_ID', all.x=TRUE, all.y = FALSE)
msri_combat <- merge(msri_combat, bss_vars, by = 'Study_ID', all.x=TRUE, all.y = FALSE)
msri_combat <- merge(msri_combat, audit_vars, by = 'Study_ID', all.x=TRUE, all.y = FALSE)

#Transform data to use mean score rather than total
msri_combat$MSRIfam <- msri_combat$MSRIfam/7
msri_combat$MSRIpse <- msri_combat$MSRIpse/7
msri_combat$MSRIsui <- msri_combat$MSRIsui/7

msri_combat <- msri_combat[msri_combat$Suicide_attempt!="99",]
msri_combat <- msri_combat[-which(is.na(msri_combat$Suicide_attempt)),]

msri_combat$Suicide_attempt[msri_combat$Suicide_attempt=="2"] <- "no attempt"
msri_combat$Suicide_attempt[msri_combat$Suicide_attempt=="1"] <- "attempt"

names(msri_combat) <- c('Study_ID','Family_Connectedness',
                        'Pos_Self_Perception','MSRI_SI',
                        'Combat','Suicide_Attempt','RFL',
                        'BSS_SI','AUDIT')
```

Predicting suicide attempts from protective and risk factors:

Further Decision Tree Parameters:

Beyond choosing a method (such as information gain or gini split), we can change other parameters to control how our decision tree is constructed. For instance, the complexity parameter, or cp, helps to control the size of our tree. It is defined by learnbymarketing as *the minimum improvement in the model needed at each node* (see link if you're interested in learning more about the parameters). Here we will create multiple trees with different complexity parameters to compare their results.

```
#set seed so that results are reproducible
set.seed(123)
#split the data into testing and training sets (70/30 split)
split = sample.split(msri_combat$Suicide_Attempt, SplitRatio = 0.7)
#get training data
train = subset(msri_combat, split==TRUE)
#get testing data
test = subset(msri_combat, split==FALSE)
#for prediction:
#remove Study_ID as it is not a predictor
#remove Suicide_Attempt since that's the response we want to predict
remove_columns <- which(names(msri_combat) %in% c('Study_ID','Suicide_Attempt'))

#get decision tree
#using information gain and a low complexity parameter (cp)
tree_lowcp = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.01),
             parms = list(split = 'information'))
#get variable importance from tree with low cp
varimp_lowcp <- tree_lowcp$variable.importance

#get decision tre
#get tree with slightly higher cp
tree_highcp = rpart(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
             method = "class", control = rpart.control(cp=.03),
             parms = list(split = 'information'))
#get variable importance from tree with higher cp
varimp_highcp <- tree_highcp$variable.importance

#predict the results of the test set using tree (information gain)
tree_highcp.pred = predict(tree_highcp, newdata=test[,-remove_columns], type='class')
#information gain, high cp
tree_lowcp.pred = predict(tree_lowcp, newdata=test[,-remove_columns], type='class')
#information gain, low cp

#to display results of tree with high CP
#summary(tree_highcp)
```

Evaluating the prediction results

```
#get a confusion matrix to view the results of the information gain prediction
x <- confusionMatrix(as.factor(tree_highcp.pred), as.factor(test$Suicide_Attempt))
#store the results as a data frame to easily access variables
#note: reference refers to ground truth
pred_results<- data.frame(x$table)

#get values for precision and recall
TP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="attempt",
             'Freq']
FP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
```

```
TN <- pred_results[pred_results$Prediction=="no attempt" &
                   pred_results$Reference=="no attempt",
                 'Freq']
FN <- pred_results[pred_results$Prediction=="no attempt" &
                   pred_results$Reference=="attempt",
                 'Freq']

#print the prediction accuracy: (TP+TN) / (TP + FP + TN + FN)
accuracy <- (TP+TN) / (TP + FP + TN + FN)
paste0("prediction accuracy: ",round(accuracy,digits = 2))
```

```
## [1] "prediction accuracy: 0.75"
```

```
#Precision: TP / (TP + FP)
precision <- TP/(TP+FP)
paste0("precision: ",round(precision,digits = 2))
```

```
## [1] "precision: 0.68"
```

```
#Recall: TP / (TP + FN)
recall <- TP/(TP+FN)
paste0("recall: ",round(recall,digits = 2))
```

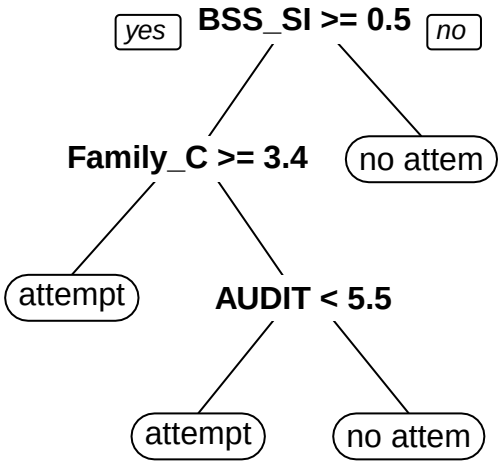
```
## [1] "recall: 0.39"
```

```
#print confusion matrix
kable(x$table, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	attempt	no attempt
attempt	17	8
no attempt	27	88

How is the decision tree classifying these participants?

```
prp(tree_highcp)
```



What can we learn about variable importance from information gain?

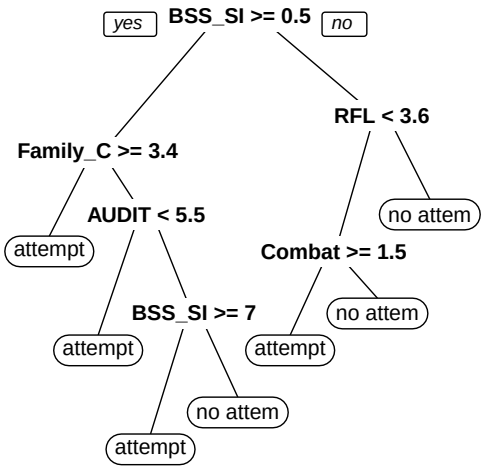
From the original tree (complexity parameter=.03):

```
a <- as.data.frame(varimp_highcp)
names(a) <- c("information gain")

kable(a, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	information gain
BSS_SI	34.620368
MSRI_SI	12.770278
Pos_Self_Perception	11.880289
RFL	9.286434
Family_Connectedness	6.158439
AUDIT	3.720848

What would this look like if we decreased the complexity parameter? That is, if we included more factors even if they did not have as big of an impact on the response variable as the best predictors?

```
b <- as.data.frame(varimp_lowcp)
names(b) <- c("variable importance")

prp(tree_lowcp)
```




```
kable(b, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	variable importance
BSS_SI	37.499282
RFL	17.088185
MSRI_SI	14.879026
Pos_Self_Perception	14.128071
Family_Connectedness	8.216749
Combat	5.152574
AUDIT	4.440576

Improving Predictive Power: Random Forests

While decision trees are helpful, we can gain even more predictive power by combining the results of many decision trees in what is called a *random forest*.

From *Towards Data Science*:

- The random forest algorithm works by aggregating the predictions made by multiple decision trees of varying depth. Every decision tree in the forest is trained on a subset of the dataset called the bootstrapped dataset.
- Some samples are left out during the construction of each decision tree in the forest, referred to as the Out-Of-Bag (OOB) dataset. The model will automatically evaluate its own performance by running each of the samples in the OOB dataset through the forest.
- By default, the number of decision trees in the forest is 500 and the number of features used as potential candidates for each split is 3.

To learn more about random forests, visit Towards Data Science

```
rf <- randomForest(as.factor(Suicide_Attempt) ~ ., data=train[,-1],
                   method = "class", control = rpart.control(cp=.01),
              parms = list(split = 'information'))
rf.pred <- predict(rf, newdata=test[,-remove_columns])
x <- confusionMatrix(as.factor(test$Suicide_Attempt), as.factor(rf.pred))

#store the results as a data frame to easily access variables
#note: reference refers to ground truth
pred_results<- data.frame(x$table)

#get values for precision and recall
TP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="attempt",
             'Freq']
FP <- pred_results[pred_results$Prediction=="attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
TN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="no attempt",
             'Freq']
FN <- pred_results[pred_results$Prediction=="no attempt" &
               pred_results$Reference=="attempt",
             'Freq']

#print the prediction accuracy: (TP+TN) / (TP + FP + TN + FN)
accuracy <- (TP+TN) / (TP + FP + TN + FN)
paste0("prediction accuracy: ",round(accuracy,digits = 2))
```

```
## [1] "prediction accuracy: 0.74"
```

```
#Precision: TP / (TP + FP)
precision <- TP/(TP+FP)
paste0("precision: ",round(precision,digits = 2))
```

```
## [1] "precision: 0.39"
```

```
#Recall: TP / (TP + FN)
recall <- TP/(TP+FN)
paste0("recall: ",round(recall,digits = 2))
```

```
## [1] "recall: 0.65"
```

```
#print confusion matrix
kable(x$table, "latex", booktabs=T)%>%
  row_spec(0,bold=FALSE) %>%
  kable_styling(latex_options="hold_position",font_size = 10)
```

	attempt	no attempt
attempt	17	27
no attempt	9	87

Conclusion

The best way to improve predictive power is not only to construct a random forest full of decision trees, but to tune the forest's hyperparameters such as the number of trees in the forest, the complexity of the trees, and the features used in the model. Keep in mind that these outcomes depend on many factors such as

- how the data was cleaned (e.g. what was done with the missing variables?)
- what type of decision criteria were used (e.g. gini index, information gain)
- the hyperparameters (e.g. the complexity parameter of a tree, the number of trees in a forest)
- feature selection and transformation (what if we had used different predictor variables? What if we had used the log of those values?)

Further Hypothesis Testing:

Hypothesis: Scores on the problematic drinking (AUDIT) scale and the Family Connectedness scale will predict scores on the Beck Scale for Suicide Ideation (BSSI) for each group separately.

```
ms3 <- MSRI[,c("Study_ID","MSRIfam")]
ad1 <-Audit[,c("Study_ID","AUDITtot")]

ms_ad <- inner_join(ms3, ad1, by="Study_ID")

bs1 <-BSS[,c("Study_ID","BSS19_Total")]

pred_bss <- inner_join(ms_ad, bs1, by="Study_ID")

c_group <- osb_gen[,c("Study_ID","Combat")]
c_group$Combat[c_group$Combat==1] <- 0
c_group$Combat[c_group$Combat==2] <- 1

pred_bss <- left_join(pred_bss, c_group, by="Study_ID")


reg_tot <- lm(data=pred_bss, BSS19_Total~MSRIfam+AUDITtot+Combat)

reg_tot
```

```
##
## Call:
## lm(formula = BSS19_Total ~ MSRIfam + AUDITtot + Combat, data = pred_bss)
##
## Coefficients:
## (Intercept)      MSRIfam     AUDITtot       Combat
##     4.93205     -0.12230     -0.02502     -0.85455
```

```
summary(reg_tot)
```

```
##
## Call:
## lm(formula = BSS19_Total ~ MSRIfam + AUDITtot + Combat, data = pred_bss)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0009 -1.6611 -0.8846  0.0821 20.0010
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.93205    0.61873   7.971 1.19e-14 ***
## MSRIfam     -0.12230    0.01877  -6.515 1.87e-10 ***
## AUDITtot    -0.02502    0.06707  -0.373  0.70925
## Combat      -0.85455    0.32374  -2.640  0.00857 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.478 on 472 degrees of freedom
## Multiple R-squared:  0.09035,    Adjusted R-squared:  0.08456
```

```
## F-statistic: 15.63 on 3 and 472 DF,  p-value: 1.05e-09
```

We can see that combat group has a significant effect on Beck Scale for Suicide Ideation total scores.

```
pred_bss1 <- pred_bss[pred_bss$Combat==0,]
pred_bss2 <- pred_bss[pred_bss$Combat==1,]

reg1 <- lm(data=pred_bss1, BSS19_Total~MSRIfam+AUDITtot)
reg2 <- lm(data=pred_bss2, BSS19_Total~MSRIfam+AUDITtot)
```

For combat group:

```
reg1
```

```
##
## Call:
## lm(formula = BSS19_Total ~ MSRIfam + AUDITtot, data = pred_bss1)
##
## Coefficients:
## (Intercept)      MSRIfam     AUDITtot
##     6.37073     -0.17429     -0.05999
```

For non-combat group:

```
reg2
```

```
##
## Call:
## lm(formula = BSS19_Total ~ MSRIfam + AUDITtot, data = pred_bss2)
##
## Coefficients:
## (Intercept)      MSRIfam     AUDITtot
##     3.30565     -0.09041     -0.01374
```