DATA 622: Homework #4

Karim Hammoud and Michael Munguia

11/12/2021

Section 1

Conduct a thorough Exploratory Data Analysis (EDA) to understand the dataset.

We can began our exploration of the data by first viewing summary statistics for each numeric variable with a call to the `summary` function. Rather than reproduce the lengthy output here, the outcome was an awareness of where missing values occurred in the data. A subset of `summary` output on the data set can be seen below:

```
summary(select(adhd, Opioids, Abuse, `Subst Dx`, `Psych meds.`))
```

```
##     Opioids           Abuse          Subst Dx      Psych meds.
##  Min.   :0.0000   Min.   :0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.000   Median :1.000   Median :1.0000
##  Mean   :0.3918   Mean   :1.329   Mean   :1.138   Mean   :0.9649
##  3rd Qu.:0.0000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.0000
##  Max.   :3.0000   Max.   :7.000   Max.   :3.000   Max.   :2.0000
##  NA's   :4        NA's   :14      NA's   :23      NA's   :118
```

The various substance-use related variables show missing values similar to the counts we see in `Opioids`/`Abuse`, which on first glance do not seem critical but occur in such disparate observations that we can quickly reduce our overall data set from its already small original 175 observations. The original observation-count is not a huge amount to work with, so we want to be very cautious when developing our models down the road that we are choosy with when and where we drop missing values.

We also want to consider what variables we omit wholesale to avoid issues incurred from missing values. For example, `Psych meds.` is relatively useless as 67% of these values are missing. We can assume this to be a zero-value, but that's a much more risky assumption than, say, doing the same for `Opioids`.

We'll create an `explore` function to streamline visualization of the data set, as shown below:

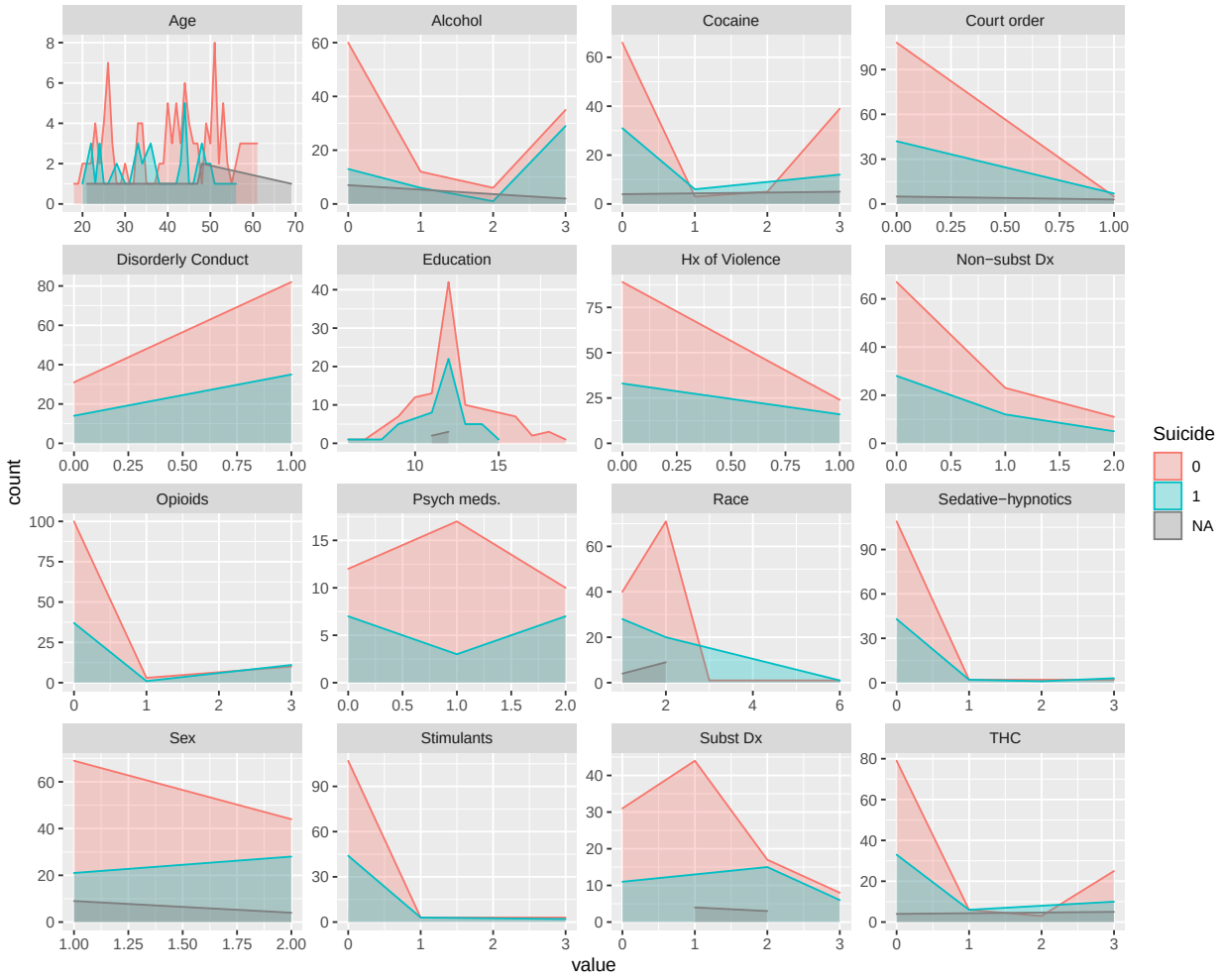
```
adhd_cols <- str_subset(colnames(adhd), "^ADHD")
md_cols <- str_subset(colnames(adhd), "^MD")

demo_cols <- colnames(adhd)[!colnames(adhd) %in% c(adhd_cols, md_cols)]
demo_cols <- demo_cols[!demo_cols %in% c("Index", "Initial", "Suicide")]

explore <- function(columns, stat, n_cols = 4) {
  adhd %>%
```

```
    pivot_longer(cols = all_of(columns)) %>%
    mutate(across("Suicide", factor)) %>%
    ggplot(aes(x = value, color = Suicide, fill = Suicide, group = Suicide)) +
    geom_density(alpha = 0.3, stat = stat) +
    facet_wrap(~ name, scales = "free", ncol = n_cols)
}
```

There are three broad categories of variables recorded in the data - general intake data and two sets coming from an ADHD self-report and mood disorders (MD) questionnaire. The general intake data includes substance-usage and demographic data.

We will visualize the distribution of responses within subsets in succession, starting with the non-questionnaire variables, below. Because we will want to eventually predict suicide attempts with this data, emphasis is placed on understanding the data through that lens.

```
explore(demo_cols[demo_cols != "Abuse"], stat = "count")
```



There are a few interesting takeaways from this subset of the data. There seems to be no true pattern across ages, though one might argue there are more older patients reporting no suicide attempts - which is intuitive,

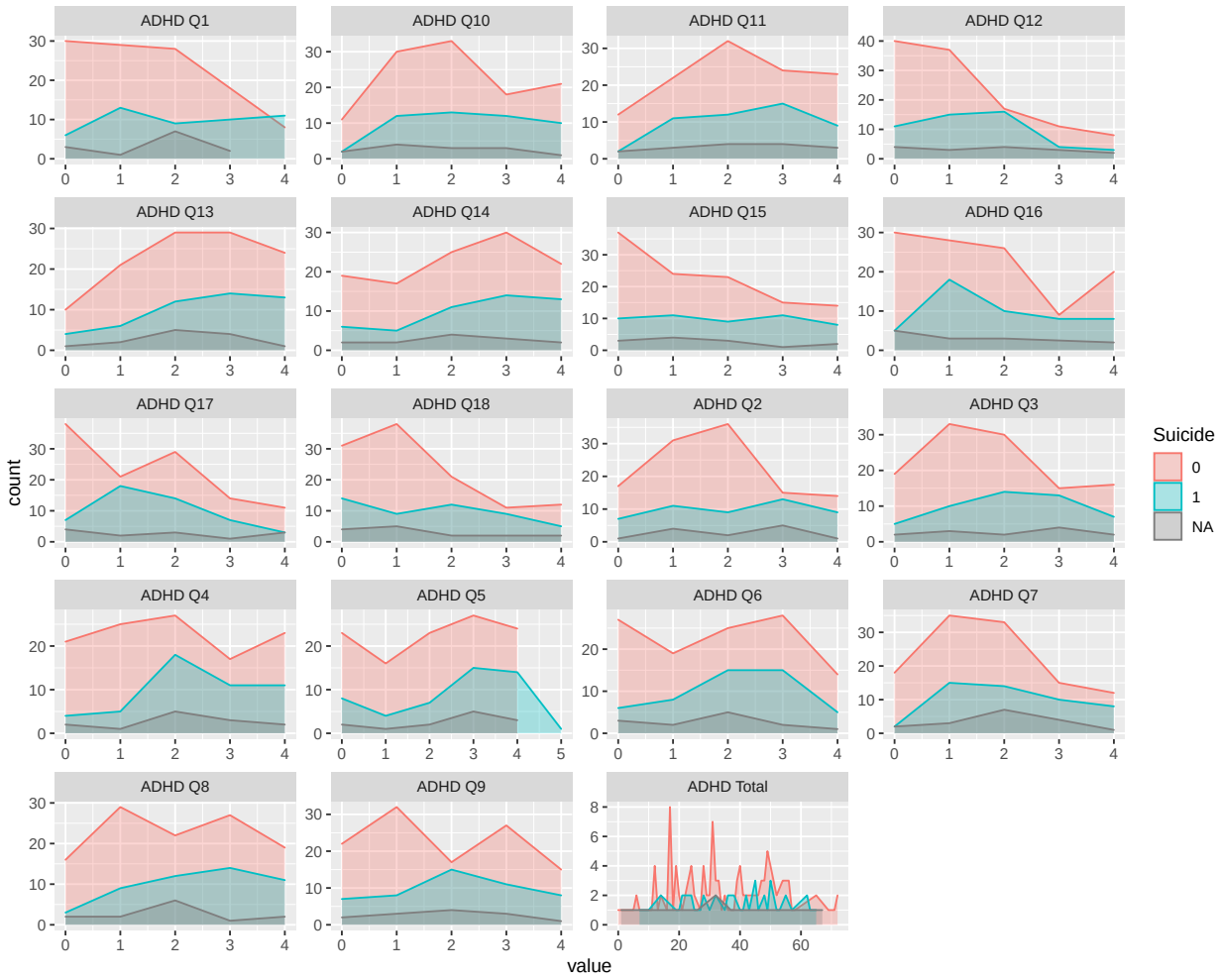
as younger suicidal patients would have a higher likelihood of not reaching an old age than their non-suicidal peers.

Variables measuring a history of substance use across a spectrum of abstinence to dependence show a decrease in prevalence of non-suicidal patients the further they veer towards substance dependence. Neither cocaine nor THC have as a pronounced a change in this regard as other substances.

It initially seems like psychiatric medication self-reporting might be insightful, however, 67% of patients lack a response for this variable so it is ultimately unhelpful. Sex appears to have a relationship with suicide attempts in the sample with more men than women having never attempted suicide. More women in the sample appear to have made a suicide attempt than not with the percentage being about 42% and 57% for men and women respectively. Race may be an important factor, but the data set is heavily inbalanced and representative of black (100) and white (72) patients out of the overall 175 patients.

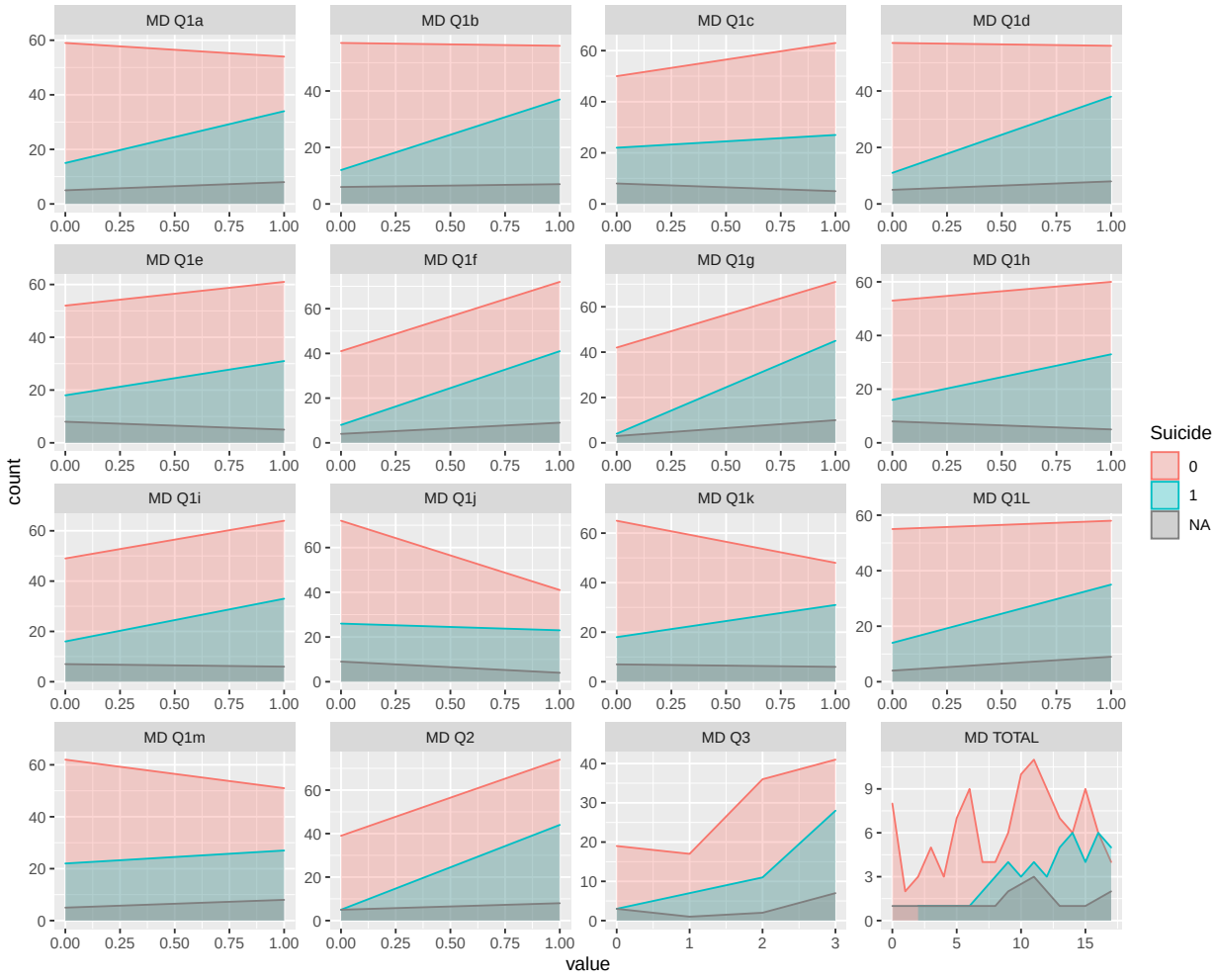
Visualizing the results from the ADHD questionnaire are difficult to summarize without the context of the question-content itself, but we can see that certain questions seemed to illicit higher responses from either group. Unlike the results from the MD assessment, there isn't as clear a throughline from question-to-question. Broadly speaking, a higher score in the assessment seems to come with higher incidences of suicide attempt.

```
explore(adhd_cols, stat = "count")
```



Visualizing the responses from the MD questionnaire reveal that in most cases there is a pronounced increase in the proportion of patients having attempted suicide pending a "Yes" answer to the given question.

```
explore(md_cols, stat = "count")
```



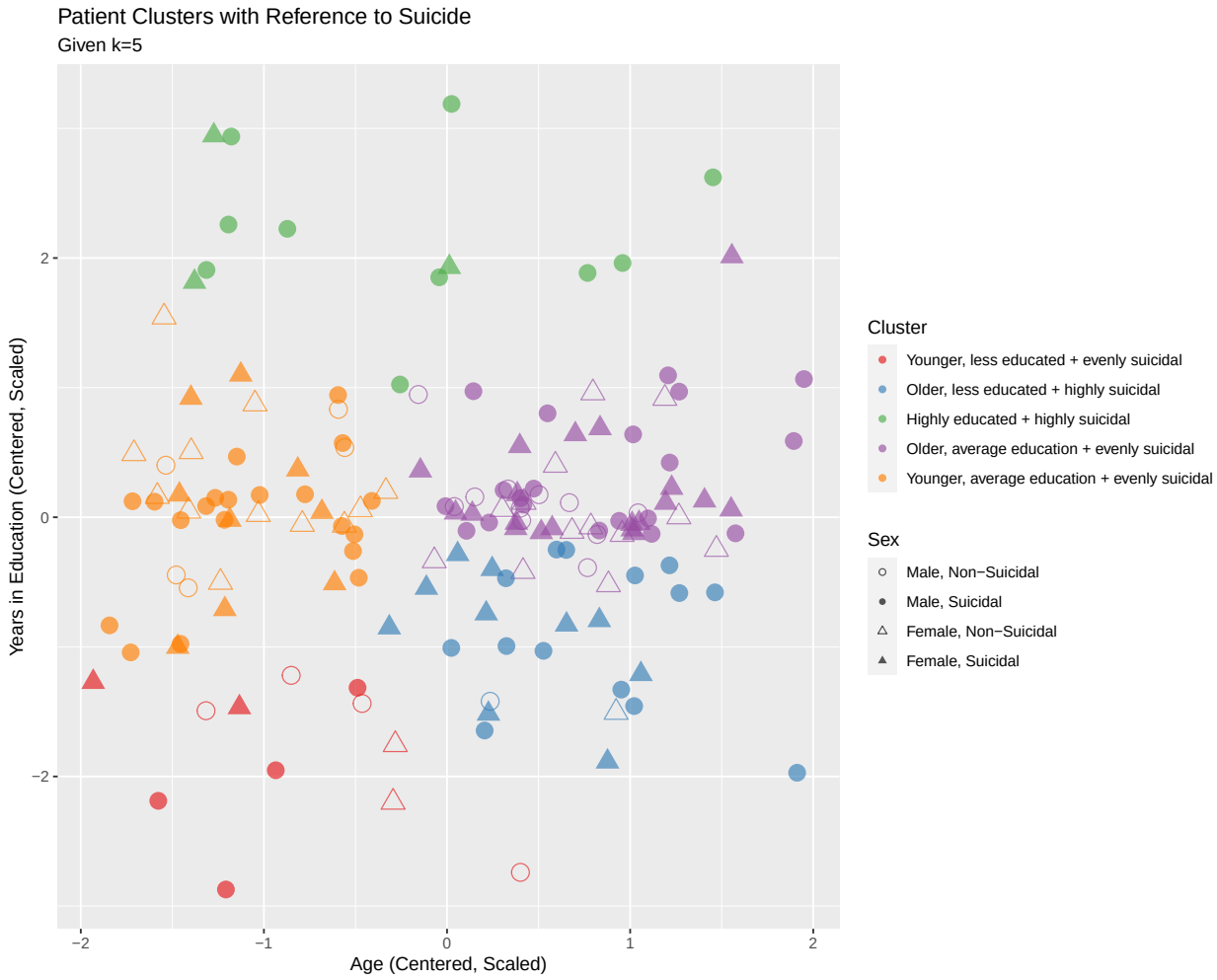
Section 2

> Use a clustering method to find clusters of patients here. Whether you choose to use k-means clustering or hierarchical clustering is up to you as long as you reason through your work. You are free to be creative in terms of which variables or some combination of those you want to use. Can you come up with creative names for the profiles you found?

We ran through the process of finding patient clusters below, utilizing the k-Means method. It seemed the most reasonable to try and do this based on some of demographic data that was made available in the data set. To this end, we opted to use `Age`, `Education`, `Sex` and `Suicide` - the last variable included due to the overall focus of predicting suicide attempts in this assignment. Both the `Age` and `Education` variables were centered and scaled due to the fact that while consisting of the same unit, they can vary wildly as years-in-education are inevitability eclipsed by one's ongoing lifespan. This should make clustering more effective, but the scaled variables will still be quite interpretable even if they no longer are on a year-unit.

```r
set.seed(3)

kmeans_vars <- c(x = "Age", y = "Education", shape = "Sex", fill = "Suicide")
k <- 5

kmeans_df <- adhd %>%
  select(all_of(kmeans_vars)) %>%
  mutate(
    x = scale(x)[ , 1],
    y = scale(y)[ , 1],
    across(c("shape", "fill"), factor)
    ) %>%
  drop_na()

kmeans_df$cluster <- factor(kmeans(kmeans_df, centers = k)$cluster)
```

Our thinking was that because we do not have any true pre-labeling or foreknowledge as to what the clusters themselves will be, these variables are the easiest to interpret as describable groups of patients. Deciding on a value for k was a fairly subjective process and we iterated between values of 3 and 5, eventually settling on 5 as it felt the most easily interpreted. Visualizing the end result went hand in hand with interpretation, so these two steps were duly linked. The visualization's code is quite long, and so has been hidden from view in this report. Rather than list out the labels in text, they are included below as the color legend in the following visualization of the results.



Section 3

Let's explore using Principal Component Analysis on this dataset. You will note that there are different types of questions in the dataset: column: E-W: ADHD self-report; column X – AM: mood disorders questionnaire, column AN-AS: Individual Substance Misuse; etc. You could just use ONE of the sets of questionnaire, for example, you can conduct PCA on the ADHD score, or mood disorder score, etc. Please reason through your work as you decide on which sets of variables you want to use to conduct Principal Component Analysis. What did you learn from the PCA? Can you comment on which question may have a heavy bearing on the score?

We opted to utilize principal component analysis (PCA) on the ADHD questionnaire data, primarily because its constituent variables showed an interesting but difficult to characterize relationship with suicidality in patients. We select the subset of variables and conduct PCA below. The `ADHD Total` variable is removed as it is clearly dependent on all the other ADHD questionnaire variables. Because these variables all follow the exact same scoring scheme, we do not need to worry about centering/scaling for the purpose of PCA.

```
adhd_pca <- adhd %>%
  select(starts_with("ADHD"), -`ADHD Total`) %>%
  prcomp()
```

In doing so, we can review the `summary` output for the `prcomp` object below. It handily lines up each principal component (PC) and displays the standard deviation, proportion of variance and the running/cumulative proportion of the same. The initial takeaway here is that we can explain *half* of the variance using the first PC alone. If we used the first five we would have nearly 74% of the variance explained with only five variables versus the original eighteen we started with.

```
summary(adhd_pca)
```

```
## Importance of components:
##                            PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.9503 1.5390 1.33031 1.16038 1.09870 1.00804 0.95612
## Proportion of Variance 0.5152 0.0782 0.05843 0.04445 0.03985 0.03355 0.03018
## Cumulative Proportion  0.5152 0.5934 0.65183 0.69628 0.73613 0.76968 0.79986
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.91331 0.86183 0.84714 0.80830 0.79002 0.76129 0.69338
## Proportion of Variance 0.02754 0.02452 0.02369 0.02157 0.02061 0.01913 0.01587
## Cumulative Proportion  0.82740 0.85192 0.87562 0.89719 0.91779 0.93693 0.95280
##                           PC15    PC16    PC17    PC18
## Standard deviation     0.68103 0.60581 0.56805 0.52549
## Proportion of Variance 0.01531 0.01212 0.01065 0.00912
## Cumulative Proportion  0.96811 0.98023 0.99088 1.00000
```

We can access a matrix of the relative impact of each input variable on the output PCs through the `rotation` attribute on the `prcomp` object. We slice this matrix below to view just the first five PCs that get us to 74% explanatory power.

```
adhd_pca$rotation[ , 1:5]
```

```
##                PC1          PC2         PC3         PC4           PC5
## ADHD Q1  0.2267492 -0.154796252  0.35974787 -0.39454560  0.1824374852
## ADHD Q2  0.2230954 -0.237844959  0.25302831 -0.24061554  0.3468360921
## ADHD Q3  0.2154347 -0.298669459  0.11084011 -0.02816598 -0.4983774010
## ADHD Q4  0.2494255 -0.306369017  0.22506736 -0.05512727  0.0099911028
## ADHD Q5  0.2679741 -0.172506867 -0.51422311  0.03780035 -0.2949790068
## ADHD Q6  0.2153420  0.014417734 -0.35365427 -0.46787827 -0.1156216308
## ADHD Q7  0.2200585 -0.114828732  0.07590637 -0.03060760 -0.2093611006
## ADHD Q8  0.2618820 -0.113403467  0.12656655  0.26629056 -0.1402511002
## ADHD Q9  0.2651426 -0.030983123  0.10787878  0.29068675  0.0147218732
## ADHD Q10 0.2462733 -0.054215833  0.09289959  0.11055794  0.0498552070
## ADHD Q11 0.2309140 -0.144647135 -0.08590211  0.22581252  0.2449729014
## ADHD Q12 0.2084479  0.166240379  0.11700582  0.34624952 -0.1257143481
## ADHD Q13 0.2384432 -0.038909358 -0.36449614  0.04384759  0.2731356258
## ADHD Q14 0.2468546 -0.007360681 -0.35264096  0.04563835  0.4274638673
## ADHD Q15 0.2284996  0.379064517 -0.02231018 -0.34948655 -0.2406362650
## ADHD Q16 0.2370968  0.499936796  0.14706789 -0.09822870  0.1624198468
## ADHD Q17 0.2134244  0.335788450  0.07858240  0.28545493 -0.0001807051
## ADHD Q18 0.2355961  0.346386093  0.07407429 -0.07441489 -0.1180061080
```

The first PC shows that each question essentially holds the same degree of impact in its composition and that initial 50% of explanatory ability. Going down the line, however, we start to see that certain questions shift in their overall impact as we bridge the gap between 50% and 74%. For example, compared to the other questions, Q1 and Q2 show a consistent degree of impact across our first five PCs and the same could be said to a lesser extent for Q5. This guides us in thinking that these three questions are particularly strong in their overall impact on a patient's score.

Section 4

> Assume you are modeling whether a patient attempted suicide (column AX). This is a binary target variable. Please use Gradient Boosting to predict whether a patient attempts suicides. Please use whatever boosting approach you deem appropriate. But please be sure to walk us through your steps.

In the following sections, we will follow conduct predictive modeling with both a gradient boosting based model (GBM) and a support vector machine based model (SVM) to predict suicide attempts. Even though it is included in the prompt for the next section, we will do some feature engineering here so that both models can benefit from that work. There were a few observations during data exploration that informed this process, which revolved around creating a new series of binary variables valued 0-1 to either reduce the variables utilized or simplify them into a tidier format. Our logic is summarized as follows:

- Despite the lack of diversity in the data set, the `Race` variable reveals a higher proportion of white participants have attempted suicide than black participants. The "Hispanic" label is also a broad heritage rather than a specific race, making it inappropriate to record here. Because there were so few participants labeled as something other than black or white, this variable is reduced to the binary variable `White`.
- Because all the substance-use variables measured the same behavior (i.e. a spectrum of non-user, user and addict) these could each be reduced to a binary representing whether the participant is simply a user of the given controlled substance. Missing values could essentially be seen as the participant being unwilling to self-report their usage - a behavior unlikely for non-users and so these could be imputed as `0`. We experimented with this as a single case-statement driven variable (i.e. a single variable for usage of any substances), but keeping each substance separate seemed to help the models better generalize.
- The `Abuse` variable was converted to a tidy variable that represents whether a participant self-reported as the victim of any kind of abuse. Contrasting to the substance-abuse variables, this seemed to produce better results as a single catch-all.
- The two assessments had their many variables reduced to a pair of performance scores based on the maximum possible scores eluded to in the data dictionary.

In order to setup our model, the prepped data needed to be sampled. Because the number of suicidal versus non-suicidal participants was imbalanced (about 28% where `Suicide = 1`), we setup the training set to include a more deliberate sampling across both groups. This does not result in a perfect 50/50 split - that would not be an unreasonable expectation given this data set. It does, however, provide a better set of inputs than if we naively sampled the entire data set to reserve 20% of it for testing.

```
adhd2 <- adhd %>%
  mutate(
    White = if_else(Race == 1, 1, 0),

    Alcohol = if_else(Alcohol != 0, 1, 0),
    THC = if_else(THC != 0, 1, 0),
    Cocaine = if_else(Cocaine != 0, 1, 0),
    Stimulants = if_else(Stimulants != 0, 1, 0),
    `Sedative-hypnotics` = if_else(`Sedative-hypnotics` != 0, 1, 0),
    Opioids = if_else(Opioids != 0, 1, 0),

    Victim = if_else(`Abuse` != 0, 1, 0),

    ADHD = `ADHD Total` / 72,
```

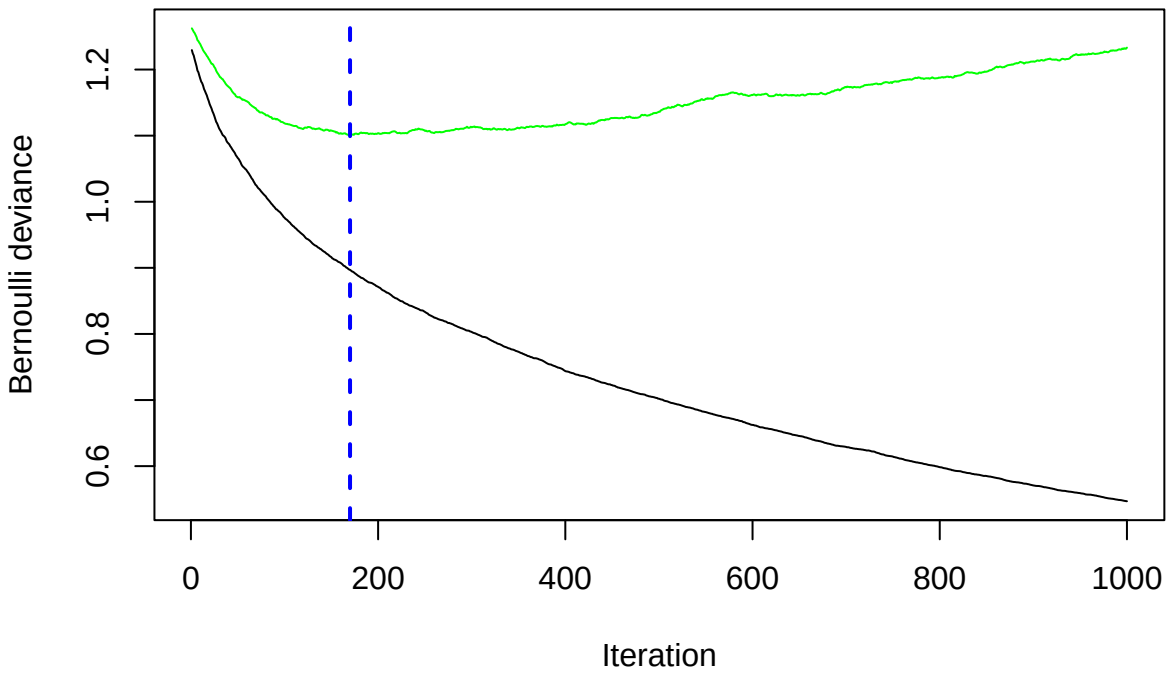
```
    MD = `MD TOTAL` / 17
  ) %>%
  select(
    Index,
    Suicide,
    White,
    Alcohol, THC, Cocaine, Stimulants, `Sedative-hypnotics`, Opioids,
    Victim,
    ADHD,
    MD
    ) %>%
  drop_na()
```

We configure the model to ignore the `Index` variable created for cross-referencing observations and choose Bernoulli as our method via the `distribution` parameter given the target variables 0/1 outcome. This will provide us with as the target variable is a binary outcome. We setup 10-fold cross validation as well, and establish that we will 1,000 separate iterations. The `interaction.depth` parameter specifies the depth of the trees were generate in the process where `interaction.depth = 1` would be a stump.

```
library(gbm)

set.seed(3)

training <- bind_rows(
  slice_sample(filter(adhd2, Suicide == 0), prop = .80),
  slice_sample(filter(adhd2, Suicide == 1), prop = .80)
  )
testing <- anti_join(adhd2, training, by = "Index")

gbm_mod <- gbm(
  Suicide ~ . -Index,
  distribution = "bernoulli",
  data = training,
  n.trees = 1000,
  interaction.depth = 3,
  shrinkage = 0.01,
  cv.folds = 10,
)
```

The `gbm` library provides a method for us to output the best performing number of trees/iterations based on our cross-validation. We store this number below with the side effect of viewing the results of the iterative process.

```
best_gbm <- gbm.perf(gbm_mod)
```



The prediction output we generate below will not arrive immediately as the output class. For that, we create a threshold and apply it to the prediction output to assign a class, much like with logistic regression via `glm`.

```
gbm_train_pred <- predict(
  gbm_mod, newdata = select(training, -Suicide),
  n.trees = best_gbm,
  type = "response"
  )

gbm_test_pred <- predict(
  gbm_mod, newdata = select(testing, -Suicide),
  n.trees = best_gbm,
  type = "response"
  )

threshold <- 0.40

training$GBM <- as.double(gbm_train_pred > threshold)
testing$GBM <- as.double(gbm_test_pred > threshold)
```

After defining a confusion matrix function, we can produce the results below for both the training and test data sets.

```
conf_mat <- function(df, prediction_column) {
  table("expected" = df[["Suicide"]], "predicted" = df[[prediction_column]])
}
```

```
conf_mat(training, "GBM")
```

```
##          predicted
## expected  0  1
##        0 83  5
##        1 18 21
```

```
conf_mat(testing, "GBM")
```

```
##          predicted
## expected  0  1
##        0 20  3
##        1  9  1
```

The training and testing results are 81% and 63% accuracy respectively. While initially promising, this does not seem like a model that generalizes particularly well to new data. Further, this particular model also has a tendency to predict false negatives in its misclassifications. In the following section, however, we will the result of another type of model that will we will favor for this particular prediction task.

Section 5

> Using the same target variable (suicide attempt), please use support vector machine to model this. You might want to consider reducing the number of variables or somehow use extracted information from the variables. This can be a really fun modeling task!

The `e1071` library, chosen for this task for its familiarity, requires the target variable be stored as a factor in this case and so we apply that additional transformation atop our earlier data preparation.

With a training data set in hand, we can pass a formula for our model to the `best.svm` method and have it fit an optimal SVM-based model given our situation. The resulting `C-classification` type and `radial` kernel are selected as a result - this accords with general results from manual experimentation we conducted outside of this report.

We exclude both the `Index` and earlier `GBM` prediction variables.

```
library(e1071)

training <- mutate(training, Suicide = factor(Suicide))
testing <- mutate(testing, Suicide = factor(Suicide))

set.seed(3)

best_svm <- best.svm(
  x = Suicide ~ . -Index -GBM,
  data = training
)

best_svm
```

```
##
## Call:
## best.svm(x = Suicide ~ . - Index - GBM, data = training)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  87
```

Given the model above, we can generate some predictions using the usual `predict` method for SVM models and store that on our data sets. This allows us to easily create confusion matrices and evaluate the results.

```
##          predicted
## expected  0  1
##        0 85  3
##        1 16 23
```

```
conf_mat(testing, "SVM")
```

```
##          predicted
## expected  0  1
##        0 20  3
##        1  5  5
```

Based on the confusion matrices for the training and testing data respectively displayed above, we calculate about 85% and 75% accuracy respectively. This model performs comparably as the earlier GBM on training data, but requires significantly less computation time. It also seems to do a better job generalizing to new data as there is a far less stark drop in accuracy when utilizing the testing data.

As would be anticipated, there is a drop in accuracy between training and testing data but in having experimented with other kernel options, it looks like the optimal model provided by the `best.svm` method generalizes relatively well by comparison.

It does, unfortunately, seem to have a training false negative rate of about 41% in training and about 50% in testing. Not knowing the eventually goal/outcome for the model's usage, a model with a lower false negative rate and higher false positive rate might be preferable if one is wishing to be cautious in targeting suicidal patients for intervention/treatment.