---

title: "Sex Differences Profiling: SOM followed by MClust"

output:

 html\_document:

 keep\_md: yes

 number\_sections: yes

 toc: yes

 toc\_float: yes

editor\_options:

 chunk\_output\_type: console

---

```{r Load\_Packages, warning=FALSE, include=FALSE}

library(readxl)

library(ggplot2)

library(reshape2)

library(kohonen)

library(ggplot2)

library(factoextra)

library(xlsx)

library(mclust)

library(dplyr)

library(arsenal)

library(htmlTable)

library(tidyr)

library(ggthemes)

library(tidyverse)

library(broom)

library(dlookr)

library(kableExtra)

library(summarytools)

library(qwraps2)

library(finalfit)

library(plotly)

library(tableone)

library(compareGroups)

library(htmltools)

library(data.table)

library(gridExtra)

library(caret)

library(caretEnsemble)

library(DMwR)

library(mice)

library(VIM)

library(pROC)

##### functions to be used later

createAndPlotClustersSOM <- function(DataToCluster, clustVars, DataInfo){

 s=1234

 CompleteScaledData <- as.matrix(scale(DataToCluster[,clustVars]))

 #Create SOM Grid

 som\_grid <- somgrid(xdim = 10, ydim=10, topo="hexagonal", neighbourhood.fct = "gaussian")

 #som\_grid <- somgrid(xdim = 10, ydim=10, topo="hexagonal") # for smaller sample sizes ()

 #Train SOM

 set.seed(s)

 som\_model <- som(CompleteScaledData,

 grid=som\_grid,

 rlen=200,

 #alpha=c(0.05,0.01),

 keep.data = TRUE)

 SOMCodedata <- as.data.frame(som\_model$codes )

 set.seed(s)

 MClustered <- Mclust(SOMCodedata, G=3:20)

 m.best2 <- dim(MClustered$z)[2]

 cat("model-based optimal number of clusters:", m.best2, "\n")

 fvizplot<- fviz\_nbclust(SOMCodedata, kmeans, method = "wss", k.max = 20)

 #plot(fvizplot)

 #Now do the actual clustering

 som\_cluster<-MClustered$classification

 patient\_SOM = som\_model$unit.classif

 patient\_clust = som\_cluster[patient\_SOM]

 ClusterData<-as.data.frame(DataToCluster[,cogVars])

 ClusterData$Cluster <- as.factor(patient\_clust)

 ClusterData$ID<-seq.int(nrow(ClusterData))

 DataToCluster$Cluster<- ClusterData$Cluster

 #count summary of what is in each and add to cluster names

 summary(ClusterData$Cluster)

 s<-summary(ClusterData$Cluster)

 clusternames<-paste(1:length(s), '(n=', s,')' )

 e<-mean(1-MClustered$uncertainty)

 ti<- paste(DataInfo, 'n =', length(patient\_clust), 'model = ', MClustered$modelName, ',Entropy =', round(e, digits = 3))

 mdata <-melt(ClusterData, id = c('ID', 'Cluster'))

 p <- ggplot(mdata, aes(Cluster, value, fill = variable)) + geom\_boxplot() +

 ylab("Value")+scale\_fill\_discrete(name = 'Test')+

 geom\_hline(yintercept=45, linetype = "dashed", size = .5)+geom\_hline(yintercept=55, linetype = "dashed", size = .5)+

 geom\_hline(yintercept=40, linetype = "dashed", size = 1)+geom\_hline(yintercept=60, linetype = "dashed", size = 1)+

 scale\_x\_discrete(labels = clusternames) + ggtitle(ti)

 ClustResults = list("SomCodeddata" = SOMCodedata, "MClustered" = MClustered, "ClusterData" = DataToCluster, "p" = p)

 return(ClustResults)

}

# calculate the best cluster

calcTopCluster <- function(ClusteredDat, cogVars){

 #Calculate the Mean of each of the cog vars for each cluster

#find the cluster that has the highest average of the means

 dat<- ClusteredDat[c('Cluster', cogVars)]

 s<- with(dat, aggregate(dat, by = list(Cluster), FUN = mean))

 sm = c()

 for(i in 1:nlevels(ClusteredDat$Cluster)){

 sm[i]<- mean(t(s[i, cogVars]))

 }

 topCluster = levels(ClusteredDat$Cluster)[sm==max(sm)]

 return(topCluster)

}

createRFModel <- function(ModelData, normalClusters){

 s = 123456

 clustList<-levels(ModelData$Cluster)

 ClustersOfInterest<-clustList[!(clustList %in% normalClusters)]

 #return(ClustResults)

 MR = list() #Model Results

for (c in 1:length(ClustersOfInterest)){

 ClusterofInterest<-ClustersOfInterest[c]

 CI<-ClusterofInterest

 MR[[CI]]$NormalData<-ModelData[ModelData$Cluster %in% normalClusters,]

 MR[[CI]]$NormalData$Cluster<-'Normal'

 MR[[CI]]$ClusterData<-ModelData[ModelData$Cluster == ClusterofInterest,]

 MR[[CI]]$DataSubset<-rbind(MR[[CI]]$ClusterData , MR[[CI]]$NormalData)

 MR[[CI]]$DataSubset$Cluster<-as.factor(as.character(MR[[CI]]$DataSubset$Cluster))

 MR[[CI]]$DataSubset<-as.data.frame(MR[[CI]]$DataSubset)

 set.seed(s)

 inTraining<-createDataPartition(MR[[CI]]$DataSubset$Cluster, p = 0.70, list = FALSE)

 MR[[CI]]$training<-MR[[CI]]$DataSubset[inTraining,]

 MR[[CI]]$testing<-MR[[CI]]$DataSubset[-inTraining,]

 #Smote Resampling

 set.seed(s)

 MR[[CI]]$smote\_train <- SMOTE(Cluster ~ ., data = MR[[CI]]$DataSubset, perc.over=100)

 tunegrid <- expand.grid(.mtry=c(1:ncol(ModelData)))

 fitControl <- trainControl(method = "repeatedcv",

 number = 10,

 repeats = 5,

 classProbs = TRUE,

 allowParallel = TRUE,

 summaryFunction = twoClassSummary

 )

 data = MR[[CI]]$smote\_train

 set.seed(s)

 MR[[CI]]$FIT\_RandomForest<-train( data[colnames(data)[2:length(colnames(data))]],data$Cluster, method = "rf" , ntrees = 5000, trControl = fitControl, na.action = na.pass, importance = TRUE, metric = 'ROC', seed = s)

}

 #Compare Normal to All (Make This The first column of the variable importance plot)

'%!in%' <- function(x,y)!('%in%'(x,y))

CI<-'ImpairedProfile'

ClusterofInterest<-CI

MR[[CI]]$NormalData<-ModelData[ModelData$Cluster %in% normalClusters,]

MR[[CI]]$NormalData$Cluster<-'Normal'

MR[[CI]]$ClusterData<-ModelData[ModelData$Cluster %!in% normalClusters,]

MR[[CI]]$ClusterData$Cluster<-'ImpairedProfile'

MR[[CI]]$DataSubset<-rbind(MR[[CI]]$ClusterData , MR[[CI]]$NormalData)

MR[[CI]]$DataSubset$Cluster<-as.factor(as.character(MR[[CI]]$DataSubset$Cluster))

MR[[CI]]$DataSubset<-as.data.frame(MR[[CI]]$DataSubset)

set.seed(123)

inTraining<-createDataPartition(MR[[CI]]$DataSubset$Cluster, p = 0.70, list = FALSE)

MR[[CI]]$training<-MR[[CI]]$DataSubset[inTraining,]

MR[[CI]]$testing<-MR[[CI]]$DataSubset[-inTraining,]

#summary(MR[[CI]]$training)

#Smote Resampling

set.seed(1234)

MR[[CI]]$smote\_train <- SMOTE(Cluster ~ ., data = MR[[CI]]$DataSubset, perc.over=100)

tunegrid <- expand.grid(.mtry=c(1:ncol(ModelData)))

fitControl <- trainControl(method = "repeatedcv",

 number = 10,

 repeats = 5,

 classProbs = TRUE,

 allowParallel = TRUE,

 summaryFunction = twoClassSummary

)

data = MR[[CI]]$smote\_train

set.seed(1234)

s = 1234

MR[[CI]]$FIT\_RandomForest<-train( data[colnames(data)[2:length(colnames(data))]],data$Cluster, method = "rf" , ntrees = 5000, trControl = fitControl, na.action = na.pass, importance = TRUE, metric = 'ROC', seed = s)

 return(MR)

}

```

<style>

 .superbigimage{

 overflow-x:scroll;

 white-space: nowrap;

 overflow-y:scroll;

 max-height: 1000px;

 }

 .superbigimage img{

 max-width: none;

</style>

# Cogntive Data Summary

```{r Load\_Data, include=TRUE, warning = FALSE, echo = FALSE, fig.width = 8}

print("SOM using 10x10 grid, hexagonal topography, gaussian function. MClust 3:20")

AllData <- read\_excel("C:/Users/RahaD/Dropbox/Pilot Grant/Sex Difference Profile/Sundermann UCSD Sex difference Profile ML.xlsx")

cogVars = c('tbttotc\_t\_pe', 'tbdtrl\_t\_pe','BVMTRecog\_AgeEducSexRaceNormTscore', 'hpttotc\_t\_pe', 'hpdtrl\_t\_pe', 'HVLTRecog\_AgeEducSexRaceNormTscore', 'mtgdtsr\_n\_pe', 'mtgntscr\_n\_pe','tratscr\_n\_pe','trbtscr\_n\_pe','ltftscr\_n\_pe','pc5tscr\_n\_pe','dytscr\_n\_pe')

cogVarlabels = c('BVMT Learning','BVMT Delayed Recall','BVMT Recognition','HVLT Learning','HVLT Delayed Recall','HVLT Recognition','Pegs Dominant','Pegs Non-dominant','Trails A','Trails B','FAS','PASAT 50','WAIS III Digit Symbol')

predictorVars<- c('bnviage', 'max\_education', 'WRAT\_last', 'WhtBlkHisOther', 'IADLComplaints', 'BDI\_combined', 'LTMDD\_R', 'CurrentMDD\_R', 'LTorCurrAlcoholDx','LTorCurrCannabisDx' ,'LTorCurrSubsDx\_NotETOHorMJ', 'Hypertension\_R', 'Hyperlipidemia\_R', 'Diabetes\_R', 'CurrentAnticholinergicMeds\_R', 'HCV\_R', 'rnlog10PLASMA', 'NadirCD4', 'CD4Absolute', 'EstimatedDurationofInfectionyears', 'ARTonoff' )

factorVars<- c('WhtBlkHisOther','LTMDD\_R', 'CurrentMDD\_R', 'LTorCurrAlcoholDx','LTorCurrCannabisDx', 'LTorCurrSubsDx\_NotETOHorMJ' ,'Hypertension\_R', 'Hyperlipidemia\_R', 'Diabetes\_R', 'CurrentAnticholinergicMeds\_R', 'HCV\_R', 'ARTonoff' )

#filter data by completeness

labels(AllData[cogVars])= cogVarlabels

Data = AllData[c('linknum', cogVars)]

Data = AllData[cogVars]

#label(Data)= cogVarlabels

paste('Started with', nrow(Data), 'subjects')

AllData<- AllData[complete.cases(Data),]

labels(AllData[cogVars])= cogVarlabels

Data = AllData[cogVars]

#Plot histograms of everything - point out outliers

#Set any value<0 to 0 and any value>100 to 100

Data[Data<0]<- 0

Data[Data>100]<- 100

Data$ID = seq(1, nrow(Data))

df = melt(Data, id = "ID")

p<- ggplot(df, aes(x = value, alpha = 0.5)) + geom\_density(aes(fill = variable)) + facet\_wrap(~variable, scales = "free")

p<- p+ scale\_fill\_brewer(palette = "Paired", labels = cogVarlabels)

plot(p)

#filter by individual values

#Data <- Data%>% filter(tbttotc\_t\_pe>0,tbdtrl\_t\_pe >0, hpttotc\_t\_pe >0, hpdtrl\_t\_pe >0, mtgdtsr\_n\_pe >0,mtgntscr\_n\_pe >0, tratscr\_n\_pe >0,

# trbtscr\_n\_pe>0 , ltftscr\_n\_pe >0, pc5tscr\_n\_pe >0, BVMTRecog\_AgeEducSexRaceNormTscore>-20, HVLTRecog\_AgeEducSexRaceNormTscore>-20,

 # dytscr\_n\_pe >0)

#

paste('Ended with', nrow(Data), 'subjects with complete data')

CompleteData<- AllData

CompleteData[cogVars]<- Data[cogVars]

```

## Descriptive Summaries by Gender {.tabset .tabset-fade}

### Data Table

```{r descriptive\_summary\_All, warning = FALSE, echo=FALSE, include=TRUE}

numVars<-predictorVars[which(!predictorVars %in% factorVars)]

catVars<- c('bngendr', factorVars)

#Make sure that All Categorical Variables are factors

ModelData<- CompleteData[c('bngendr', cogVars, predictorVars)]

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

tab1<-tableby(bngendr~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("Nmiss2","meansd"), chisq.correct = TRUE, cat.test = "chisq")

tabplain<- tableby(bngendr~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"), text = TRUE)

sd<-as.data.frame(summary(tab1, text = FALSE))

sd2<-as.data.frame(summary(tab1), text = TRUE)

colnames(sd)[length(colnames(sd))]<-'p\_value'

pvalscript<- sd$p\_value

pvalscript[pvalscript == "< 0.001"]<- "0"

pvals<-as.numeric(pvalscript)

pvals[is.na(pvals)]<-1

colnames(sd)[1]<-'Variable'

colnames(sd2)[1]<-'Variable'

t<- tests(tab1)

options(kableExtra.auto\_format = T)

#colnames(sd)[length(colnames(sd))]<-p\_value

#bold the first column of anything that has a p-value

#haspvalue<- sd$`p value`!=""

sd<-as.data.table(sd)

sd%>% mutate(p\_value = cell\_spec(p\_value , "html", background = ifelse(pvals<0.05, "yellow", "")))%>%dplyr::rename("p-value" = p\_value)%>%

 kable(format = "html", escape = F)%>% kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"))%>%

 scroll\_box(width = "100%", height = "500px")

```

### Continuous Data Figures

<div class="superbigimage">

```{r Continuous\_Fig\_All, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "bngendr"

yvars <- c(cogVars, numVars)

ModelData<-CompleteData[,c(xvar, yvars)]

tab1<- tableby(bngendr~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"))

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(xvar, var)])

 testn <-paste(colnames(NewData)[[2]],pvalscript[i], sep = '\n' )

 colnames(NewData)<-c("bngendr", "value")

 NewData$bngendr<- as.factor(NewData$bngendr)

 pl<-ggplot(NewData, aes(x = bngendr, y = value))

 pl<- pl + geom\_boxplot(width = 0.2)

 pl<- pl + geom\_violin(aes(fill = bngendr), draw\_quantiles = .5, na.rm = TRUE, alpha = 0.5)

 pl<-pl+scale\_fill\_brewer(palette = "Set3") + ggtitle(testn) + theme\_clean()

 pl<- pl+theme(text = element\_text(size = 15), axis.title.y = element\_blank(), axis.title.x = element\_blank(),axis.text.x = element\_text(angle = 45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

#plot(pl)

 plotList[[i]]<-pl

#multiplot(plotlist = plotList, cols = 4)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

### Categorical Data Figures

<div class="superbigimage">

```{r Grouped\_Stats\_Cat\_Fig\_All, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "bngendr"

yvars <- catVars[!catVars %in% xvar]

ModelData<-CompleteData[,c(xvar, yvars)]

ModelData<-as.data.frame(lapply(ModelData, factor))

#colnames(ModelData)<-labels(ModelData)

tab1<-tableby(as.formula(paste(xvar, "~ .")),ModelData)

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(paste(xvar), var)])

 testn <-paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' ) #paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' )

 ##calculate percentage

 #plot percentage

 NewData <- NewData %>%

 group\_by\_(xvar, var) %>%

 summarise(count=n()) %>%

 mutate(Percent=count/sum(count))

 NewData$Percent<- NewData$Percent\*100

 p<-ggplot(NewData, aes\_string(x =xvar, y = 'Percent')) + ggtitle(testn)

 p<- p+geom\_bar(stat = "identity",aes\_string(fill=var))

 p<- p+ scale\_fill\_brewer(palette = "Dark2") + theme\_clean()

 p<- p+ theme( text = element\_text(size = 15), axis.title.y = element\_blank(), axis.text.x = element\_text(angle =45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

 #pl

 plotList[[i]] <- p

 # HTMLplot(GraphRes = 500)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

## Cluster Results

```{r Cluster\_All, warning = FALSE, echo=FALSE,include=TRUE, fig.width = 8}

ClusterData\_All<- CompleteData

DataInfo = 'All Participants'

AllClusteredList<- createAndPlotClustersSOM(ClusterData\_All, cogVars, DataInfo)

p<- AllClusteredList$p

p<- p+ scale\_fill\_brewer(palette = "Paired", labels = cogVarlabels)

plot(p)

```

## Cluster Statistics{.tabset .tabset-fade}

### Data Table by Cluster

```{r Table\_All, warning = FALSE, echo=FALSE, include=TRUE}

numVars<-predictorVars[which(!predictorVars %in% factorVars)]

catVars<- c('Cluster', 'bngendr', factorVars)

#Make sure that All Categorical Variables are factors

ModelData<-AllClusteredList$ClusterData[c('Cluster', 'bngendr', cogVars, predictorVars)]

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

tab1<-tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("Nmiss2", "meansd"), cat.test = "chisq")

tabplain<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"), text = TRUE)

sd<-as.data.frame(summary(tab1, text = FALSE))

sd2<-as.data.frame(summary(tab1), text = TRUE)

colnames(sd)[length(colnames(sd))]<-'p\_value'

pvalscript<- sd$p\_value

pvalscript[pvalscript == "< 0.001"]<- "0"

pvals<-as.numeric(pvalscript)

pvals[is.na(pvals)]<-1

colnames(sd)[1]<-'Variable'

colnames(sd2)[1]<-'Variable'

t<- tests(tab1)

options(kableExtra.auto\_format = T)

#colnames(sd)[length(colnames(sd))]<-p\_value

#bold the first column of anything that has a p-value

#haspvalue<- sd$`p value`!=""

sd<-as.data.table(sd)

sd%>% mutate(p\_value = cell\_spec(p\_value , "html", background = ifelse(pvals<0.05, "yellow", "")))%>%dplyr::rename("p-value" = p\_value)%>%

 kable(format = "html", escape = F)%>% kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"))%>%

 scroll\_box(width = "100%", height = "500px")

```

<div class="tocify-extend-page" data-unique="tocify-extend-page" style="height: 0;"></div>

### Continuous Data Figures

<div class="superbigimage">

```{r Continuous\_Fig\_ClustAll, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- c(cogVars, numVars)

ModelData<-AllClusteredList$ClusterData[,c(xvar, yvars)]

tab1<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"))

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(xvar, var)])

 testn <-paste(colnames(NewData)[[2]],pvalscript[i], sep = '\n' )

 colnames(NewData)<-c("Cluster", "value")

 NewData$Cluster<- as.factor(NewData$Cluster)

 pl<-ggplot(NewData, aes(x = Cluster, y = value))

 pl<- pl + geom\_boxplot(width = 0.2)

 pl<- pl + geom\_violin(aes(fill = Cluster), draw\_quantiles = .5, na.rm = TRUE, alpha = 0.5)

 pl<-pl+scale\_fill\_brewer(palette = "Set3") + ggtitle(testn) + theme\_clean()

 pl<- pl+theme(text = element\_text(size = 15), axis.title.y = element\_blank(), axis.title.x = element\_blank(),axis.text.x = element\_text(angle = 45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

#plot(pl)

 plotList[[i]]<-pl

#multiplot(plotlist = plotList, cols = 4)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

### Categorical Data Figures

<div class="superbigimage">

```{r Cat\_Fig\_All, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- catVars[!catVars %in% xvar]

ModelData<-AllClusteredList$ClusterData[,c(xvar, 'bngendr',yvars)]

ModelData<-as.data.frame(lapply(ModelData, factor))

#colnames(ModelData)<-labels(ModelData)

tab1<-tableby(as.formula(paste(xvar, "~ .")),ModelData)

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(paste(xvar), var)])

 testn <-paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' ) #paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' )

 ##calculate percentage

 #plot percentage

 NewData <- NewData %>%

 group\_by\_(xvar, var) %>%

 summarise(count=n()) %>%

 mutate(Percent=count/sum(count))

 NewData$Percent<- NewData$Percent\*100

 p<-ggplot(NewData, aes\_string(x =xvar, y = 'Percent')) + ggtitle(testn)

 p<- p+geom\_bar(stat = "identity",aes\_string(fill=var))

 p<- p+ scale\_fill\_brewer(palette = "Dark2") + theme\_clean()

 p<- p+ theme( text = element\_text(size = 15), axis.title.y = element\_blank(), axis.text.x = element\_text(angle =45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

 #pl

 plotList[[i]] <- p

 # HTMLplot(GraphRes = 500)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

## Random Forest Model Creation: All Data

```{r RF\_All, warning = FALSE, echo=FALSE,include=TRUE}

#figure out which clusters are normal and which arent

ModelData\_All<- AllClusteredList$ClusterData

catVars<- c('Cluster', 'bngendr', factorVars)

#Make sure that All Categorical Variables are factors

ModelData\_All[catVars]<- lapply(ModelData\_All[catVars],as.character)

ModelData\_All[catVars] <- lapply(ModelData\_All[catVars] , factor)

ModelData\_All[catVars]<- lapply(ModelData\_All[catVars], make.names)

#if there's a dot, remove it

ModelData<- as.data.frame(ModelData\_All[c('Cluster', 'bngendr', predictorVars)])

ModelData[catVars][ModelData[catVars]=="NA."] <- NA

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

ModelData\_All[catVars] <- lapply(ModelData\_All[catVars] , factor)

topClust = calcTopCluster(ModelData\_All, cogVars)

paste("The normal cluster is detected as being:", topClust)

#Remove Variables before runing through RF

all\_f = sapply(ModelData[,catVars], function(x) nlevels(x)>1)

removed\_onelevelfactors <- names(all\_f[all\_f==FALSE])

paste('The following Variables were removed because they had less than 2 factors: ', toString(removed\_onelevelfactors))

ModelData<-select(ModelData, -removed\_onelevelfactors)

#No go through and look for missing variables

aggr\_plot\_all <- aggr(ModelData, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"), bars = TRUE, labels = TRUE, plot = FALSE)

thresh = .5 \* nrow(ModelData)

remove\_ind = aggr\_plot\_all$missings$Count>thresh

removed\_30pmissing<-colnames(ModelData[,remove\_ind])

paste('The following Variables were removed because they had more than 50% missing: ', toString(removed\_30pmissing))

ModelData<-select(ModelData, -removed\_30pmissing)

lowVarPredictors = nearZeroVar(ModelData, saveMetrics = TRUE)

removed\_lowVar<- rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]

paste('The following Variables were removed because they had near zero variance: ', toString(rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]))

ModelData<-select(ModelData, rownames(lowVarPredictors)[lowVarPredictors$nzv==FALSE])

paste('Final Variables: ', toString(colnames(ModelData)))

#impute the other variables

imputed\_temp <-mice(ModelData, meth = "rf", seed = 123, printFlag = FALSE)

ModelData\_imputed\_all = mice::complete(imputed\_temp)

RF\_all<- createRFModel(ModelData\_imputed\_all, topClust)

remove('df', 'imputed\_temp', 'lowVarPredictors', 'ModelData',' NewData', 'p', 'pl', 'plotList', 'sd', 'sd2', 't', 'tab1', 'tabplain', 'i', 'remove\_ind', 'remove\_30pmissing', 'stripcolor', 'testn', 'thresh', 'var', 'topClust')

```

## All: Random Forest Results {.tabset .tabset-fade}

### Variable Importance

```{r All\_RF\_VI, warning = FALSE, echo=FALSE, include=TRUE}

##Calculate and Plot Variable Importance Using different sorting methods

MR<- RF\_all

groups<- attributes(MR)$names

CIs<- groups[1:length(groups)-1]

if(length(CIs)==1){

 CIs <- "ImpairedProfile"

}else{

CIs<-c('ImpairedProfile',CIs)

}

for (c in 1:length(CIs)){

 ClusterofInterest<-CIs[c]

 CI<-ClusterofInterest

 Model\_RF <- MR[[CI]]$FIT\_RandomForest

 model = Model\_RF

 model\_label<- CI

 imps = varImp(model, scale = TRUE)

 if(c==1){

 Imp\_df = data.frame(row.names = predictors(model))

Imp\_df$ID <-seq.int(nrow(Imp\_df))

Imp\_df$Variable <-rownames(Imp\_df)

Imp\_df$Variable<-factor(Imp\_df$Variable, levels = rev(Imp\_df$Variable) )

 }

 imps2 = imps[["importance"]]

 if (ncol(imps2) ==2){

 imps2 = imps2[ClusterofInterest]

 }

 colnames(imps2) = model\_label

 imps2$Variable = rownames(imps2)

 Imp\_df<-merge(Imp\_df, imps2,by = "Variable", all = TRUE )

}

newdf<-Imp\_df[order(Imp\_df$ImpairedProfile),]

newdf$ID <-seq.int(nrow(newdf))

#newdf$Variable<- VarTypes1[[2]][match(newdf$Variable, VarTypes1[[1]])]

dfImp<-melt(newdf, id = c('ID', 'Variable') )

#Rename Variables

##Try highlighing top value

me.2 <- dfImp %>%

 group\_by(variable) %>% mutate(color = value %in% head(sort(value, decreasing = TRUE),10) )

p<- ggplot(me.2, aes(x = Variable, y = value)) +coord\_flip()

p<- p + geom\_segment(size = 2,aes(x = Variable, xend = Variable, y =0, yend = value, color = color )) + scale\_x\_discrete(limits = newdf$Variable)

p<-p + facet\_wrap(~variable, nrow=1) +theme(legend.position = "none", strip.text.y = element\_text(angle=45), axis.title.x = element\_blank(), axis.title.y = element\_blank())

p<- p + geom\_point() +ylab('Variable Importance') + ggtitle('Random Forest Variable Importance') +scale\_color\_manual(values = c("lightgrey", "black"))

plot(p)

```

### ROC Plots

```{r All\_RF\_ROC, warning = FALSE, echo=FALSE, include=TRUE}

##Plot ROC

for (c in 1:length(CIs)){

 CI<-CIs[c]

 #Plot ROCs

 model <- MR[[CI]]$FIT\_RandomForest

 Pred\_RF <-predict(model, MR[[CI]]$DataSubset, type = "prob")

 ROC\_obj<-roc(response = MR[[CI]]$DataSubset$Cluster, predictor = Pred\_RF[,CI], ci = TRUE)

 ci.sp.obj <- ci.sp(ROC\_obj, sensitivities=seq(0, 1, .01))

 plot(ROC\_obj)

 plot(ci.sp.obj, type="shape", col="lightblue")

 title(CI, line = 2)

 print(model)

 print(model$finalModel)

}

remove('dfImp', 'Imp\_df', 'imps', 'imps2', 'me.2', 'model', 'model\_RF', 'modelData\_imputed', 'MR', 'NewData', 'newdf', 'Pred\_RF', 'RF', 'ROC\_obj', 'all\_f', 'c', 'CI', 'CIs', 'groups', 'MatchingVars', 'model\_label', 'namedVector', 'pvalscript', 'removed\_30pmissing', 'removed\_lowVar', 'removed\_onelevelfactors', 'p' )

```

# Males Only

## Cluster Results

```{r Cluster\_Men, warning = FALSE, echo=FALSE,include=TRUE, fig.width = 8}

ClusterData\_Men<- CompleteData[CompleteData$bngendr==1,]

DataInfo = 'Males'

MenClusteredList<- createAndPlotClustersSOM(ClusterData\_Men, cogVars, DataInfo)

p<- MenClusteredList$p

p<- p+ scale\_fill\_brewer(palette = "Paired", labels = cogVarlabels)

plot(p)

```

## Cluster Statistics{.tabset .tabset-fade}

### Data Table by Cluster

```{r Table\_Men, warning = FALSE, echo=FALSE, include=TRUE}

numVars<-predictorVars[which(!predictorVars %in% factorVars)]

catVars<- c('Cluster', factorVars)

#Make sure that All Categorical Variables are factors

ModelData<-MenClusteredList$ClusterData[c('Cluster', cogVars, predictorVars)]

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

tab1<-tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("Nmiss2", "meansd"), cat.test = "chisq")

tabplain<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"), text = TRUE)

sd<-as.data.frame(summary(tab1, text = FALSE))

sd2<-as.data.frame(summary(tab1), text = TRUE)

colnames(sd)[length(colnames(sd))]<-'p\_value'

pvalscript<- sd$p\_value

pvalscript[pvalscript == "< 0.001"]<- "0"

pvals<-as.numeric(pvalscript)

pvals[is.na(pvals)]<-1

colnames(sd)[1]<-'Variable'

colnames(sd2)[1]<-'Variable'

t<- tests(tab1)

options(kableExtra.auto\_format = T)

#colnames(sd)[length(colnames(sd))]<-p\_value

#bold the first column of anything that has a p-value

#haspvalue<- sd$`p value`!=""

sd<-as.data.table(sd)

sd%>% mutate(p\_value = cell\_spec(p\_value , "html", background = ifelse(pvals<0.05, "yellow", "")))%>%dplyr::rename("p-value" = p\_value)%>%

 kable(format = "html", escape = F)%>% kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"))%>%

 scroll\_box(width = "100%", height = "500px")

```

### Continuous Data Figures

<div class="superbigimage">

```{r Continuous\_Fig\_ClustMen, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- c(cogVars, numVars)

ModelData<-MenClusteredList$ClusterData[,c(xvar, yvars)]

tab1<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"))

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(xvar, var)])

 testn <-paste(colnames(NewData)[[2]],pvalscript[i], sep = '\n' )

 colnames(NewData)<-c("Cluster", "value")

 NewData$Cluster<- as.factor(NewData$Cluster)

 pl<-ggplot(NewData, aes(x = Cluster, y = value))

 pl<- pl + geom\_boxplot(width = 0.2)

 pl<- pl + geom\_violin(aes(fill = Cluster), draw\_quantiles = .5, na.rm = TRUE, alpha = 0.5)

 pl<-pl+scale\_fill\_brewer(palette = "Set3") + ggtitle(testn) + theme\_clean()

 pl<- pl+theme(text = element\_text(size = 15), axis.title.y = element\_blank(), axis.title.x = element\_blank(),axis.text.x = element\_text(angle = 45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

#plot(pl)

 plotList[[i]]<-pl

#multiplot(plotlist = plotList, cols = 4)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

### Categorical Data Figures

<div class="superbigimage">

```{r Cat\_Fig\_Men, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- catVars[!catVars %in% xvar]

ModelData<-MenClusteredList$ClusterData[,c(xvar, yvars)]

ModelData<-as.data.frame(lapply(ModelData, factor))

#colnames(ModelData)<-labels(ModelData)

tab1<-tableby(as.formula(paste(xvar, "~ .")),ModelData)

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(paste(xvar), var)])

 testn <-paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' ) #paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' )

 ##calculate percentage

 #plot percentage

 NewData <- NewData %>%

 group\_by\_(xvar, var) %>%

 summarise(count=n()) %>%

 mutate(Percent=count/sum(count))

 NewData$Percent<- NewData$Percent\*100

 p<-ggplot(NewData, aes\_string(x =xvar, y = 'Percent')) + ggtitle(testn)

 p<- p+geom\_bar(stat = "identity",aes\_string(fill=var))

 p<- p+ scale\_fill\_brewer(palette = "Dark2") + theme\_clean()

 p<- p+ theme( text = element\_text(size = 15), axis.title.y = element\_blank(), axis.text.x = element\_text(angle =45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

 #pl

 plotList[[i]] <- p

 # HTMLplot(GraphRes = 500)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

## Random Forest Model Creation: Males

```{r RF\_Men, warning = FALSE, echo=FALSE,include=TRUE}

#figure out which clusters are normal and which arent

ModelData\_Men<- MenClusteredList$ClusterData

catVars<- c('Cluster', factorVars)

#Make sure that All Categorical Variables are factors

ModelData\_Men[catVars]<- lapply(ModelData\_Men[catVars],as.character)

ModelData\_Men[catVars] <- lapply(ModelData\_Men[catVars] , factor)

ModelData\_Men[catVars]<- lapply(ModelData\_Men[catVars], make.names)

#if there's a dot, remove it

ModelData<- as.data.frame(ModelData\_Men[c('Cluster', predictorVars)])

ModelData[catVars][ModelData[catVars]=="NA."] <- NA

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

ModelData\_Men[catVars] <- lapply(ModelData\_Men[catVars] , factor)

topClust = calcTopCluster(ModelData\_Men, cogVars)

paste("The normal cluster is detected as being:", topClust)

#Remove Variables before runing through RF

all\_f = sapply(ModelData[,catVars], function(x) nlevels(x)>1)

removed\_onelevelfactors <- names(all\_f[all\_f==FALSE])

paste('The following Variables were removed because they had less than 2 factors: ', toString(removed\_onelevelfactors))

ModelData<-select(ModelData, -removed\_onelevelfactors)

#No go through and look for missing variables

aggr\_plot\_all <- aggr(ModelData, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"), bars = TRUE, labels = TRUE, plot = FALSE)

thresh = .5 \* nrow(ModelData)

remove\_ind = aggr\_plot\_all$missings$Count>thresh

removed\_30pmissing<-colnames(ModelData[,remove\_ind])

paste('The following Variables were removed because they had more than 50% missing: ', toString(removed\_30pmissing))

ModelData<-select(ModelData, -removed\_30pmissing)

lowVarPredictors = nearZeroVar(ModelData, saveMetrics = TRUE)

removed\_lowVar<- rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]

paste('The following Variables were removed because they had near zero variance: ', toString(rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]))

ModelData<-select(ModelData, rownames(lowVarPredictors)[lowVarPredictors$nzv==FALSE])

paste('Final Variables: ', toString(colnames(ModelData)))

#impute the other variables

imputed\_temp <-mice(ModelData, meth = "rf", seed = 123, printFlag = FALSE)

ModelData\_imputed\_Men = mice::complete(imputed\_temp)

RF\_Men<- createRFModel(ModelData\_imputed\_Men, topClust)

remove('df', 'imputed\_temp', 'lowVarPredictors', 'ModelData',' NewData', 'p', 'pl', 'plotList', 'sd', 'sd2', 't', 'tab1', 'tabplain', 'i', 'remove\_ind', 'remove\_30pmissing', 'stripcolor', 'testn', 'thresh', 'var', 'topClust')

```

## Males: Random Forest Results {.tabset .tabset-fade}

### Variable Importance

```{r Men\_RF\_VI, warning = FALSE, echo=FALSE, include=TRUE}

##Calculate and Plot Variable Importance Using different sorting methods

MR<- RF\_Men

groups<- attributes(MR)$names

CIs<- groups[1:length(groups)-1]

if(length(CIs)==1){

 CIs <- "ImpairedProfile"

}else{

CIs<-c('ImpairedProfile',CIs)

}

for (c in 1:length(CIs)){

 ClusterofInterest<-CIs[c]

 CI<-ClusterofInterest

 Model\_RF <- MR[[CI]]$FIT\_RandomForest

 model = Model\_RF

 model\_label<- CI

 imps = varImp(model, scale = TRUE)

 if(c==1){

 Imp\_df = data.frame(row.names = predictors(model))

Imp\_df$ID <-seq.int(nrow(Imp\_df))

Imp\_df$Variable <-rownames(Imp\_df)

Imp\_df$Variable<-factor(Imp\_df$Variable, levels = rev(Imp\_df$Variable) )

 }

 imps2 = imps[["importance"]]

 if (ncol(imps2) ==2){

 imps2 = imps2[ClusterofInterest]

 }

 colnames(imps2) = model\_label

 imps2$Variable = rownames(imps2)

 Imp\_df<-merge(Imp\_df, imps2,by = "Variable", all = TRUE )

}

newdf<-Imp\_df[order(Imp\_df$ImpairedProfile),]

newdf$ID <-seq.int(nrow(newdf))

#newdf$Variable<- VarTypes1[[2]][match(newdf$Variable, VarTypes1[[1]])]

dfImp<-melt(newdf, id = c('ID', 'Variable') )

#Rename Variables

##Try highlighing top value

me.2 <- dfImp %>%

 group\_by(variable) %>% mutate(color = value %in% head(sort(value, decreasing = TRUE),10) )

p<- ggplot(me.2, aes(x = Variable, y = value)) +coord\_flip()

p<- p + geom\_segment(size = 2,aes(x = Variable, xend = Variable, y =0, yend = value, color = color )) + scale\_x\_discrete(limits = newdf$Variable)

p<-p + facet\_wrap(~variable, nrow=1) +theme(legend.position = "none", strip.text.y = element\_text(angle=45), axis.title.x = element\_blank(), axis.title.y = element\_blank())

p<- p + geom\_point() +ylab('Variable Importance') + ggtitle('Random Forest Variable Importance') +scale\_color\_manual(values = c("lightgrey", "black"))

plot(p)

```

### ROC Plots

```{r Men\_RF\_ROC, warning = FALSE, echo=FALSE, include=TRUE}

##Plot ROC

for (c in 1:length(CIs)){

 CI<-CIs[c]

 #Plot ROCs

 model <- MR[[CI]]$FIT\_RandomForest

 Pred\_RF <-predict(model, MR[[CI]]$DataSubset, type = "prob")

 ROC\_obj<-roc(response = MR[[CI]]$DataSubset$Cluster, predictor = Pred\_RF[,CI], ci = TRUE)

 ci.sp.obj <- ci.sp(ROC\_obj, sensitivities=seq(0, 1, .01))

 plot(ROC\_obj)

 plot(ci.sp.obj, type="shape", col="lightblue")

 title(CI, line = 2)

 print(model)

 print(model$finalModel)

}

remove('dfImp', 'Imp\_df', 'imps', 'imps2', 'me.2', 'model', 'model\_RF', 'modelData\_imputed', 'MR', 'NewData', 'newdf', 'Pred\_RF', 'RF', 'ROC\_obj', 'all\_f', 'c', 'CI', 'CIs', 'groups', 'MatchingVars', 'model\_label', 'namedVector', 'pvalscript', 'removed\_30pmissing', 'removed\_lowVar', 'removed\_onelevelfactors', 'p' )

```

# Females Only

## Cluster Results

```{r Cluster\_Women, warning = FALSE, echo=FALSE,include=TRUE, fig.width = 8}

ClusterData\_Women<- CompleteData[CompleteData$bngendr==2,]

DataInfo = 'Females'

WomenClusteredList<- createAndPlotClustersSOM(ClusterData\_Women, cogVars, DataInfo)

p<- WomenClusteredList$p

p<- p+ scale\_fill\_brewer(palette = "Paired", labels = cogVarlabels)

plot(p)

```

## Cluster Statistics{.tabset .tabset-fade}

### Data Table by Cluster

```{r Table\_Women, warning = FALSE, echo=FALSE, include=TRUE}

numVars<-predictorVars[which(!predictorVars %in% factorVars)]

catVars<- c('Cluster', factorVars)

#Make sure that All Categorical Variables are factors

ModelData<-WomenClusteredList$ClusterData[c('Cluster', cogVars, predictorVars)]

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

tab1<-tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("Nmiss2", "meansd"), cat.test = "chisq")

tabplain<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"), text = TRUE)

sd<-as.data.frame(summary(tab1, text = FALSE))

sd2<-as.data.frame(summary(tab1), text = TRUE)

colnames(sd)[length(colnames(sd))]<-'p\_value'

pvalscript<- sd$p\_value

pvalscript[pvalscript == "< 0.001"]<- "0"

pvals<-as.numeric(pvalscript)

pvals[is.na(pvals)]<-1

colnames(sd)[1]<-'Variable'

colnames(sd2)[1]<-'Variable'

t<- tests(tab1)

options(kableExtra.auto\_format = T)

#colnames(sd)[length(colnames(sd))]<-p\_value

#bold the first column of anything that has a p-value

#haspvalue<- sd$`p value`!=""

sd<-as.data.table(sd)

sd%>% mutate(p\_value = cell\_spec(p\_value , "html", background = ifelse(pvals<0.05, "yellow", "")))%>%dplyr::rename("p-value" = p\_value)%>%

 kable(format = "html", escape = F)%>% kable\_styling(bootstrap\_options = c("striped", "hover", "condensed", "responsive"))%>%

 scroll\_box(width = "100%", height = "500px")

```

### Continuous Data Figures

<div class="superbigimage">

```{r Continuous\_Fig\_ClustWomen, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- c(cogVars, numVars)

ModelData<-WomenClusteredList$ClusterData[,c(xvar, yvars)]

tab1<- tableby(Cluster~., data = ModelData, digits.pct = 1, digits.count = 1, numeric.stats = c("meansd"))

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(xvar, var)])

 testn <-paste(colnames(NewData)[[2]],pvalscript[i], sep = '\n' )

 colnames(NewData)<-c("Cluster", "value")

 NewData$Cluster<- as.factor(NewData$Cluster)

 pl<-ggplot(NewData, aes(x = Cluster, y = value))

 pl<- pl + geom\_boxplot(width = 0.2)

 pl<- pl + geom\_violin(aes(fill = Cluster), draw\_quantiles = .5, na.rm = TRUE, alpha = 0.5)

 pl<-pl+scale\_fill\_brewer(palette = "Set3") + ggtitle(testn) + theme\_clean()

 pl<- pl+theme(text = element\_text(size = 15), axis.title.y = element\_blank(), axis.title.x = element\_blank(),axis.text.x = element\_text(angle = 45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

#plot(pl)

 plotList[[i]]<-pl

#multiplot(plotlist = plotList, cols = 4)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

### Categorical Data Figures

<div class="superbigimage">

```{r Cat\_Fig\_Women, warning = FALSE, echo=FALSE,include=TRUE, fig.height = 30, fig.width = 8}

xvar <- "Cluster"

yvars <- catVars[!catVars %in% xvar]

ModelData<-WomenClusteredList$ClusterData[,c(xvar, yvars)]

ModelData<-as.data.frame(lapply(ModelData, factor))

#colnames(ModelData)<-labels(ModelData)

tab1<-tableby(as.formula(paste(xvar, "~ .")),ModelData)

pvals<-tests(tab1)$p

pvalscript<- paste0('p = ', round(pvals,digits = 3))

pvalscript[pvals<0.001]<- "p<0.001"

stripcolor<-ifelse(pvals<0.05, "white", "grey")

stripcolor[is.na(pvals)]<-"grey"

namedVector = setNames(pvalscript, tests(tab1)$Variable)

testn = paste(attributes(namedVector)$names, '\n',namedVector)

testn = setNames(testn, tests(tab1)$Variable)

MatchingVars <- colnames(ModelData)[2:ncol(ModelData)]

plotList <- list()

for(i in 1: length(MatchingVars)){

 var = MatchingVars[i]

 NewData = as.data.frame(ModelData[,c(paste(xvar), var)])

 testn <-paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' ) #paste(labels(NewData)[[2]],pvalscript[i], sep = '\n' )

 ##calculate percentage

 #plot percentage

 NewData <- as.data.frame(NewData) %>%

 group\_by\_(xvar, var) %>%

 dplyr::summarise(count=n()) %>%

 mutate(Percent=count/sum(count))

 NewData$Percent<- NewData$Percent\*100

 p<-ggplot(NewData, aes\_string(x =xvar, y = 'Percent')) + ggtitle(testn)

 p<- p+geom\_bar(stat = "identity",aes\_string(fill=var))

 p<- p+ scale\_fill\_brewer(palette = "Dark2") + theme\_clean()

 p<- p+ theme( text = element\_text(size = 15), axis.title.y = element\_blank(), axis.text.x = element\_text(angle =45, hjust = 1), panel.background = element\_rect(fill = stripcolor[i]))

 #pl

 plotList[[i]] <- p

 # HTMLplot(GraphRes = 500)

}

grid.arrange(grobs = plotList, ncol = 2)

```

</div>

## Random Forest Model Creation: Females

```{r RF\_Women, warning = FALSE, echo=FALSE,include=TRUE}

#figure out which clusters are normal and which arent

ModelData\_Women<- WomenClusteredList$ClusterData

catVars<- c('Cluster', factorVars)

#Make sure that All Categorical Variables are factors

ModelData\_Women[catVars]<- lapply(ModelData\_Women[catVars],as.character)

ModelData\_Women[catVars] <- lapply(ModelData\_Women[catVars] , factor)

ModelData\_Women[catVars]<- lapply(ModelData\_Women[catVars], make.names)

#if there's a dot, remove it

ModelData<- as.data.frame(ModelData\_Women[c('Cluster', predictorVars)])

ModelData[catVars][ModelData[catVars]=="NA."] <- NA

ModelData[catVars] <- lapply(ModelData[catVars] , factor)

ModelData\_Women[catVars] <- lapply(ModelData\_Women[catVars] , factor)

topClust = calcTopCluster(ModelData\_Women, cogVars)

paste("The normal cluster is detected as being:", topClust)

#Remove Variables before runing through RF

all\_f = sapply(ModelData[,catVars], function(x) nlevels(x)>1)

removed\_onelevelfactors <- names(all\_f[all\_f==FALSE])

paste('The following Variables were removed because they had less than 2 factors: ', toString(removed\_onelevelfactors))

ModelData<-select(ModelData, -removed\_onelevelfactors)

#Now go through and look for missing variables

aggr\_plot\_all <- aggr(ModelData, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"), bars = TRUE, labels = TRUE, plot = FALSE)

thresh = .5 \* nrow(ModelData)

remove\_ind = aggr\_plot\_all$missings$Count>thresh

removed\_30pmissing<-colnames(ModelData[,remove\_ind])

paste('The following Variables were removed because they had more than 50% missing: ', toString(removed\_30pmissing))

ModelData<-select(ModelData, -removed\_30pmissing)

lowVarPredictors = nearZeroVar(ModelData, saveMetrics = TRUE)

removed\_lowVar<- rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]

paste('The following Variables were removed because they had near zero variance: ', toString(rownames(lowVarPredictors)[lowVarPredictors$nzv==TRUE ]))

ModelData<-select(ModelData, rownames(lowVarPredictors)[lowVarPredictors$nzv==FALSE])

paste('Final Variables: ', toString(colnames(ModelData)))

#impute the other variables

imputed\_temp <-mice(ModelData, meth = "rf", seed = 123, printFlag = FALSE)

ModelData\_imputed\_Women = mice::complete(imputed\_temp)

RF\_Women<- createRFModel(ModelData\_imputed\_Women, topClust)

remove('df', 'imputed\_temp', 'lowVarPredictors', 'ModelData',' NewData', 'p', 'pl', 'plotList', 'sd', 'sd2', 't', 'tab1', 'tabplain', 'i', 'remove\_ind', 'remove\_30pmissing', 'stripcolor', 'testn', 'thresh', 'var', 'topClust')

```

## Females: Random Forest Results {.tabset .tabset-fade}

### Variable Importance

```{r Women\_RF\_VI, warning = FALSE, echo=FALSE, include=TRUE}

##Calculate and Plot Variable Importance Using different sorting methods

MR<- RF\_Women

groups<- attributes(MR)$names

CIs<- groups[1:length(groups)-1]

if(length(CIs)==1){

 CIs <- "ImpairedProfile"

}else{

CIs<-c('ImpairedProfile',CIs)

}

for (c in 1:length(CIs)){

 ClusterofInterest<-CIs[c]

 CI<-ClusterofInterest

 Model\_RF <- MR[[CI]]$FIT\_RandomForest

 model = Model\_RF

 model\_label<- CI

 imps = varImp(model, scale = TRUE)

 if(c==1){

 Imp\_df = data.frame(row.names = predictors(model))

Imp\_df$ID <-seq.int(nrow(Imp\_df))

Imp\_df$Variable <-rownames(Imp\_df)

Imp\_df$Variable<-factor(Imp\_df$Variable, levels = rev(Imp\_df$Variable) )

 }

 imps2 = imps[["importance"]]

 if (ncol(imps2) ==2){

 imps2 = imps2[ClusterofInterest]

 }

 colnames(imps2) = model\_label

 imps2$Variable = rownames(imps2)

 Imp\_df<-merge(Imp\_df, imps2,by = "Variable", all = TRUE )

}

newdf<-Imp\_df[order(Imp\_df$ImpairedProfile),]

newdf$ID <-seq.int(nrow(newdf))

#newdf$Variable<- VarTypes1[[2]][match(newdf$Variable, VarTypes1[[1]])]

dfImp<-melt(newdf, id = c('ID', 'Variable') )

#Rename Variables

##Try highlighing top value

me.2 <- dfImp %>%

 group\_by(variable) %>% mutate(color = value %in% head(sort(value, decreasing = TRUE),10) )

p<- ggplot(me.2, aes(x = Variable, y = value)) +coord\_flip()

p<- p + geom\_segment(size = 2,aes(x = Variable, xend = Variable, y =0, yend = value, color = color )) + scale\_x\_discrete(limits = newdf$Variable)

p<-p + facet\_wrap(~variable, nrow=1) +theme(legend.position = "none", strip.text.y = element\_text(angle=45), axis.title.x = element\_blank(), axis.title.y = element\_blank())

p<- p + geom\_point() +ylab('Variable Importance') + ggtitle('Random Forest Variable Importance') +scale\_color\_manual(values = c("lightgrey", "black"))

plot(p)

```

### ROC Plots

```{r Women\_RF\_ROC, warning = FALSE, echo=FALSE, include=TRUE}

##Plot ROC

for (c in 1:length(CIs)){

 CI<-CIs[c]

 #Plot ROCs

 model <- MR[[CI]]$FIT\_RandomForest

 Pred\_RF <-predict(model, MR[[CI]]$DataSubset, type = "prob")

 ROC\_obj<-roc(response = MR[[CI]]$DataSubset$Cluster, predictor = Pred\_RF[,CI], ci = TRUE)

 ci.sp.obj <- ci.sp(ROC\_obj, sensitivities=seq(0, 1, .01))

 plot(ROC\_obj)

 plot(ci.sp.obj, type="shape", col="lightblue")

 title(CI, line = 2)

 print(model)

 print(model$finalModel)

}

remove('dfImp', 'Imp\_df', 'imps', 'imps2', 'me.2', 'model', 'model\_RF', 'modelData\_imputed', 'MR', 'NewData', 'newdf', 'Pred\_RF', 'RF', 'ROC\_obj', 'all\_f', 'c', 'CI', 'CIs', 'groups', 'MatchingVars', 'model\_label', 'namedVector', 'pvalscript', 'removed\_30pmissing', 'removed\_lowVar', 'removed\_onelevelfactors', 'p' )

```

<div class="tocify-extend-page" data-unique="tocify-extend-page" style="height: 0;"></div>

# Version Info

```{r PrintAndSave, warning = FALSE, echo=TRUE, include=FALSE,cache=TRUE, fig.width = 10, fig.height = 18}

filename<- paste0("SexDiffEnvirSOM3to10clust\_", format(Sys.time(), "%Y-%m-%d %I-%p"), ".RData")

save.image(file = filename)

print(paste("For future reference, the analyzed data shown here has been saved as", filename))

print(sessionInfo())

```