The results below are generated from an R script.

```
#Ocelot CaSB 185
#this code generates the simiulated clinical data used in the project

#create a dataframe
final_DF_v2 <- data.frame(matrix(ncol = 24, nrow = 0))
#add table headings
tableHeadings <- c("ICU", "Age", "Sex", "Survival Probability", "Score", "Kidney", "Blood", "Cholesterol
colnames(final_DF_clustered)  <- tableHeadings

## Error in colnames(final_DF_clustered) <- tableHeadings:  object 'final_DF_clustered' not
found

#function to generate a random age from empirical distribution of ICU patients
Select_Age_Gp <- function()
{
  x <- as.integer(sample(c(1:10), size = 1, prob=c(0.076, 0.071,0.128,0.208,0.121,0.105,0.118,0.103,0.05
  return(x)
}

#function to generate a random age from empirical distribution of ICU patients
Select_Gender <- function()
{
  return(sample(c("M", "F"), size = 1, prob = c(0.599,0.401)))
}

#decides whether the score is higher or lower than optimal randomly
Select_H_L <- function()
{
  return(as.integer(sample(c(1, 2), size = 1, prob = c(0.5,0.5))))
}

#returns dangerously high value for ith feature
v_h <- function(i)
{
  return(Feature_Health[[i,4]])
}

#returns dangerously low value for ith feature
v_l <- function(i)
{
  return(Feature_Health[[i,2]])
}

#returns optimal value for ith feature
v_o <- function(i)
{
  return(Feature_Health[[i,3]])
}

#cuts of non-sense values that may be drawn from distributions
truncate <- function(x)
{
  # score can't be less than 0
  #close to zero, need to divide by x in the mapping from abstract score to measurements
  if (x < 0)
    return(0.00001)
  # score can't be greater than 1
  if (x > 1)
    return(1)
  else
    return(x)
}

#table with mortailty rates by age
mortality_ICU <- data.frame(matrix(ncol = 2, nrow = 10))
tableHeadings <- c("Age", "Mortality")
colnames(mortality_ICU) <- tableHeadings
mortality_ICU$Age <- c(1:10)
mortality_ICU$Mortality[[1]] <- 9307/10122
mortality_ICU$Mortality[[2]] <- 8148/9514
mortality_ICU$Mortality[[3]] <- 13425/17120
mortality_ICU$Mortality[[4]] <- 20253/27899
mortality_ICU$Mortality[[5]] <- 11236/16144
mortality_ICU$Mortality[[6]] <- 9381/14130
mortality_ICU$Mortality[[7]] <- 9878/15754
mortality_ICU$Mortality[[8]] <- 7813/13820
mortality_ICU$Mortality[[9]] <- 3478/7280
mortality_ICU$Mortality[[10]] <- 858/2183

#table with optimal, dangerously high and dangerously low values for each parameter
Feature_Health <-  data.frame(matrix(ncol = 4, nrow = 13))
tableHeadings <- c("Features", "Low", "Optimal", "High")
colnames(Feature_Health) <- tableHeadings
Feature_Health$Features <- c("ApacheII",  "Urine", "Creatinine", "Urea", "Oxygen", "HR", "BP", "RBC", "I
Feature_Health$Low <- c(-1000,  50, 0.5, 7, 95, 60, 60,4.2, -1100, 35, 20000, 4000, -1000)
Feature_Health$Optimal <- c(0,  1400, 0.9, 13.5, 100, 80,85, 5.15, 100, 60, 253000, 7500, 1.1)
Feature_Health$High <- c(30, 2500, 5, 60, 10000, 100, 100, 6.1, 160, 95, 450000, 10500, 8)
Feature_Health<-na.omit(Feature_Health)

#defining parameters for data generation
variance <- 0.2 #the variance for top level nodes
denom <- 2 #denominator for variance of bottom level nodes
n <- log(9)/log(2) #fix the n parameter in the hill function
N <- 1000 #number of patient entries
#generate data
for (i in 1:N)
{
  #select age sex, survival probability
  Age_Gp <- Select_Age_Gp()
  Age <- switch(Age_Gp, sample(c(18:34), size = 1), sample(c(35:44), size = 1),sample(c(45:54), size = 1
  Sex <- Select_Gender()
  Health <- rnorm(1, mortality_ICU$Mortality[Age_Gp] , variance) #number of samples, mean, standard dev
  Health <- truncate(Health)

  #choose whether the patient died or not
  #0 indicates that the patient survived, 1 indicates that the patient dies
  death <- as.integer(sample(c(0, 1), size = 1, prob = c(Health,1-Health)))
```

```r
#chooe mid-level nodes from gaussian dist based on survival probability
Scores <- truncate(rnorm(1, Health, variance))
Kidney <- truncate(rnorm(1, Health, variance))
Blood <- truncate(rnorm(1, Health, variance))
Cholesterol <- truncate(rnorm(1, Health, variance))
Immune <- truncate(rnorm(1, Health, variance))

#Pick ApacheII score
j = 1
mu_H <- (((1-Scores)/Scores)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
ApacheII <- truncate_0(rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/denom))


#Pick Urine volume
#this one has extra comments, others follow similar structure
j = 2
#select if the score is too high or too low
H_L <- as.integer(Select_H_L())
#compute the expected too high measurement given abstract Kidney score
mu_H <- (((1-Kidney)/Kidney)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
#compute the expected too low measurement given abstract Kidney score
mu_L <-  v_o(j) - (((1-Kidney)/Kidney)^(1/n))*(v_o(j) - v_l(j))
#draw the measrement from gaussian dist with mean mu_H or mu_L
Urine <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/

#Pick Creatinine Level
j = 3
H_L <- as.integer(Select_H_L())
mu_H <- (((1-Kidney)/Kidney)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
mu_L <-  v_o(j) - (((1-Kidney)/Kidney)^(1/n))*(v_o(j) - v_l(j))
Creatinine <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High

#Pick Urea
j = 4
H_L <- as.integer(Select_H_L())
mu_H <- (((1-Kidney)/Kidney)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
mu_L <-  v_o(j) - (((1-Kidney)/Kidney)^(1/n))*(v_o(j) - v_l(j))
Urea <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/d


#Pick Oxygen
j = 5
mu_L <-  v_o(j) - (((1-Blood)/Blood)^(1/n))*(v_o(j) - v_l(j))
Oxygen <- truncate_0(rnorm(1, mu_L, abs(Feature_Health$Optimal[j]- Feature_Health$Low[j])/denom))

#Pick Heaert Rate
j = 6
H_L <- as.integer(Select_H_L())
mu_H <- (((1-Blood)/Blood)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
mu_L <-  v_o(j) - (((1-Blood)/Blood)^(1/n))*(v_o(j) - v_l(j))
HR <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/den

#Pick Blood Pressure
```

```
  j = 7
  H_L <- as.integer(Select_H_L())
  mu_H <- (((1-Blood)/Blood)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  mu_L <-  v_o(j) - (((1-Blood)/Blood)^(1/n))*(v_o(j) - v_l(j))
  BP <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/den

  #Pick RBC count
  j = 8
  H_L <- as.integer(Select_H_L())
  mu_H <- (((1-Blood)/Blood)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  mu_L <-  v_o(j) - (((1-Blood)/Blood)^(1/n))*(v_o(j) - v_l(j))
  RBC <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/de

  #Pick LDL level
  j = 9
  mu_H <- (((1-Cholesterol)/Cholesterol)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  LDL <- truncate_0(rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/denom))

  #Pick HDL level
  j = 10
  H_L <- as.integer(Select_H_L())
  mu_H <- (((1-Cholesterol)/Cholesterol)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  mu_L <-  v_o(j) - (((1-Cholesterol)/Cholesterol)^(1/n))*(v_o(j) - v_l(j))
  HDL <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/de

  #Pick Platelet count
  j = 11
  H_L <- as.integer(sample(c(1, 2), size = 1, prob = c(0.9,0.1)))
  mu_H <- (((1-Immune)/Immune)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  mu_L <-  v_o(j) - (((1-Immune)/Immune)^(1/n))*(v_o(j) - v_l(j))
  Platelet <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j

  #Pick WBC count
  j = 12
  H_L <- as.integer(sample(c(1, 2), size = 1, prob = c(0.5,0.5)))
  mu_H <- (((1-Immune)/Immune)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  mu_L <-  v_o(j) - (((1-Immune)/Immune)^(1/n))*(v_o(j) - v_l(j))
  WBC <- truncate_0(switch(H_L, rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/de

  #Pick INR level
  j = 13
  mu_H <- (((1-Immune)/Immune)^(1/n))*(v_h(j) - v_o(j)) + v_o(j)
  INR <- truncate_0(rnorm(1, mu_H, abs(Feature_Health$Optimal[j]- Feature_Health$High[j])/denom))


 #add the generated values to the dataframe
 newRow <- c(1,Age,Sex,Health,Scores,Kidney,Blood,Cholesterol,Immune,ApacheII,NA,Urine,Creatinine,Urea,C
 final_DF_v4_peak[nrow(final_DF_v4_peak)+1,] <- newRow
}
```

```
## Error in truncate_0(rnorm(1, mu_H, abs(Feature_Health$Optimal[j] - Feature_Health$High[j])/denom)):
could not find function "truncate_0"
```

```
#write to dataframe as a csv file the system
write.csv(final_DF_v4_peak, "C:/Users/Shaili Mathur/Documents/UCLA_Documents/Courses/CaSB185/Figs_Featur
## Error in is.data.frame(x):  object 'final_DF_v4_peak' not found

#produce figures of the mapping between abstract score and measurement
for (i in 1:13) {
  n <- log(9)/log(2)
  v_l <- Feature_Health[[i,2]]
  v_o <- Feature_Health[[i,3]]
  v_h <- Feature_Health[[i,4]]
  xl<- seq(0,v_o, (v_o - v_l)/50 )
  xh<- seq(v_o,v_h +10, (v_h - v_o)/50)
  yl <- 1 - ((-xl+v_o)^n)/((-xl+v_o)^n +(v_o - v_l)^n)
  yh <- 1 - ((xh-v_o)^n)/((xh-v_o)^n +(v_h-v_o)^n)
  x<- c(xl,xh)
  y<- c(yl,yh)
  min_x <- 0
  max_x <- max(x)
  if (Feature_Health[[i,1]] == "Oxygen")
  {
    min_x <- 80
    max_x <- 100
  }
  setwd("C:/Users/Shaili Mathur/Documents/UCLA_Documents/Courses/CaSB185/Figs_Features")
  png(paste0(as.character(Feature_Health[[i,1]]), "_Flipped.png"))
  plot( y,x, ylab = "Feature Value" , xlab = "Health", main = as.character(Feature_Health[[i,1]]), ylim
  dev.off()

  png(paste0(as.character(Feature_Health[[i,1]]), ".png"))
  plot( x,y, xlab = "Feature Value" , ylab = "Health", main = as.character(Feature_Health[[i,1]]), xlim
  dev.off()
}

## Error in setwd("C:/Users/Shaili Mathur/Documents/UCLA_Documents/Courses/CaSB185/Figs_Features"):
cannot change working directory

#generating dummy monotonic data
#to check if the random forest model was working well or not
monotonic_sim <- data.frame(matrix(ncol = 4, nrow = 0))
tableHeadings<- c("X", "Bin", "V1", "V2")
colnames(monotonic_sim)<- tableHeadings
for (i in 1:N)
{
  X <- runif(1, min = 0, max = 1)
  binary <- as.integer(sample(c(0, 1), size = 1, prob = c(1-X,X)))
  V1 <- rnorm(1, 5*X, 0.02)
  V2 <- rnorm(1, -3*X, 0.02)
  newRow <- c(X, binary, V1, V2)
  final_DF_v4_peak[nrow(final_DF_v4_peak)+1,] <- newRow
}

## Error in eval(expr, envir, enclos):  object 'final_DF_v4_peak' not found
```

The R session information (including the OS info, R version and all packages used):

```
sessionInfo()

## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## Matrix products: default
## BLAS:   /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versio
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] knitr_1.28
##
## loaded via a namespace (and not attached):
## [1] compiler_3.6.3 magrittr_1.5   tools_3.6.3    tinytex_0.20   stringi_1.4.6
## [6] highr_0.8      stringr_1.4.0  xfun_0.12      evaluate_0.14
```

```
Sys.time()
```

```
## [1] "2020-03-17 22:36:02 PDT"
```