

SS4850 Final Project

Rui Zhu

4/6/2021

Explanatory analysis

```
HCVdata <- read.csv(file = 'hcvdat0.csv')
suppressMessages(library(dplyr))
drop <- c("X")
HCVdata = HCVdata[,!(names(HCVdata) %in% drop)]
colnames(HCVdata) = c("Category", "Age", "Sex", "ALB", "ALP", "ALT", "AST",
                      "BIL", "CHE", "CHOL", "CREA", "GGT", "PROT")

# 1 and 2 for blood donor and suspect blood donor, 2,3,4 for hepatitis,
# fibrosis, cirrhosis respectively
HCVdata$Category <- factor(HCVdata$Category, labels=c(1,2,3,4,5),
                           levels=c('0=Blood Donor',
                                     '0s=suspect Blood Donor',
                                     '1=Hepatitis',
                                     '2=Fibrosis',
                                     '3=Cirrhosis'))

# 0 for male, 1 for female
HCVdata$Sex <- factor(HCVdata$Sex, levels=c("m", "f"), labels=c("0", "1"))
HCVdata=na.omit(HCVdata[c("Category", "Age", "Sex", "ALB", "ALP", "ALT", "AST",
                          "BIL", "CHE", "CHOL", "CREA", "GGT", "PROT")])

for(var in 1:13)
{
  HCVdata[,var]=as.numeric(HCVdata[,var])
}

tail(HCVdata)
```

##	Category	Age	Sex	ALB	ALP	ALT	AST	BIL	CHE	CHOL	CREA	GGT	PROT
## 608	5	52	2	39	37.0	1.3	30.4	21	6.33	3.78	158.2	142.5	82.7
## 609	5	58	2	34	46.4	15.0	150.0	8	6.26	3.98	56.0	49.7	80.6
## 610	5	59	2	39	51.3	19.6	285.8	40	5.77	4.51	136.1	101.1	70.5
## 611	5	62	2	32	416.6	5.9	110.3	50	5.57	6.30	55.7	650.9	68.5
## 612	5	64	2	24	102.8	2.9	44.4	20	1.54	3.02	63.0	35.9	71.3
## 613	5	64	2	29	87.3	3.5	99.0	48	1.66	3.63	66.7	64.2	82.0

EDA!!!

```
#class information, distribution of "category"
```

```
library(dplyr)
```

```
HCVdata %>%
```

```
  group_by(Category) %>%
```

```
  summarise(no_rows = length(Category))
```

```
## # A tibble: 5 x 2
```

```
##   Category no_rows
```

```
##   <dbl>   <int>
```

```
## 1     1     526
```

```
## 2     2       7
```

```
## 3     3     20
```

```
## 4     4     12
```

```
## 5     5     24
```

```
#mean of each variable
```

```
colMeans(HCVdata)
```

```
## Category      Age      Sex      ALB      ALP      ALT      AST
```

```
BIL
```

```
## 1.303905 47.417657 1.383701 41.624278 68.123090 26.575382 33.772835
```

```
11.018166
```

```
##      CHE      CHOL      CREA      GGT      PROT
```

```
## 8.203633 5.391341 81.669100 38.198472 71.890153
```

```
#median of each variable
```

```
apply(HCVdata,2,median)
```

```
## Category      Age      Sex      ALB      ALP      ALT      AST      BIL
```

```
##      1.00     47.00     1.00     41.90     66.20     22.70     25.70     7.10
```

```
##      CHE      CHOL      CREA      GGT      PROT
```

```
##      8.26     5.31     77.00     22.80     72.10
```

train and test split 7:3

```
## train and test set 7:3
```

```
HCVdata<-HCVdata[complete.cases(HCVdata),]
```

```
set.seed(4850)
```

```
sample <- sample.int(n = nrow(HCVdata), size = floor(.70*nrow(HCVdata)),  
replace = F)
```

```
train <- HCVdata[sample, ]
```

```
test <- HCVdata[-sample, ]
```

roc with random forest

```
#binary category
```

```
#1 for HCV, 0 for donor(regular/suspect)
```

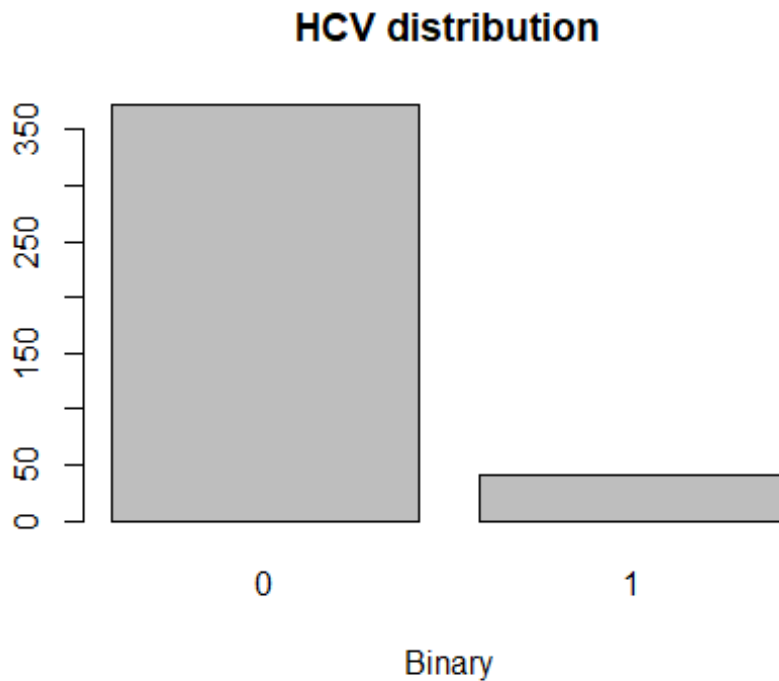
```
set.seed(4850)
```

```
train$binary <- ifelse(as.numeric(train$Category)>2,1,0)
```

```
test$binary <- ifelse(as.numeric(test$Category)>2,1,0)
```

```
#barplot on binary
```

```
counts <- table(train$binary)
barplot(counts, main="HCV distribution",
        xlab="Binary")
```



```
#Load Libraries
library(randomForest)

## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

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

rfmod <- randomForest(formula = binary ~ .-Category, data = train, ntree =
10, maxnodes= 100, norm.votes = F)

## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?

suppressMessages(library(dplyr))
testm <- mutate(test,
                 rf_pred = predict(rfmod, newdata = test,
                                   type="response"))
```

```

suppressMessages(library(pROC))

rocrf <- roc(testm$binary, testm$rf_pred)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

AUCrf <- auc(rocrf)
rs <- rocrf[['rocs']]

```

Mixed model GLMM

binary random: AST

```

suppressMessages(library(ggpubr))
library(ggplot2)
agePlot=ggplot(train, aes(x=Age,y=binary))+geom_point()
ALPPlot=ggplot(train, aes(x=ALP,y=binary))+geom_point()
ALBPlot=ggplot(train, aes(x=ALB,y=binary))+geom_point()

# regular logistic model
bmod <- glm(binary ~.-Category, binomial, data=train)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

#GLMM model random effect:Age
library(MASS)

##
## Attaching package: 'MASS'

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

set.seed(4850)
suppressMessages(modpql1 <- glmmPQL(binary
~Sex+ALB+Age+ALT+ALT+BIL+CHE+CHOL+CREA+GGT+PROT,
data=train, random= ~ 1|AST, family=binomial))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

suppressMessages(drop1(modpql1, test="Chi"))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

```

```

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Single term deletions
##
## Model:
## binary ~ Sex + ALB + Age + ALT + ALT + BIL + CHE + CHOL + CREA +
##      GGT + PROT
##      Df AIC LRT Pr(>Chi)
## <none>
## Sex      1
## ALB      1
## Age      1
## ALT      1
## BIL      1
## CHE      1
## CHOL     1
## CREA     1
## GGT      1
## PROT     1

testm1 <- mutate(test,
                  glmm_pred = predict(modpql1, newdata = test,
                                     type="response"))

suppressMessages(library(pROC))

roc1 <- roc(testm1$binary, testm1$glmm_pred)

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

AUC <- auc(roc1)
rs <- roc1[['rocs']]
coords(roc1, "best", ret=c("threshold",
                          "specificity", "sensitivity", "accuracy", "precision"))

##      threshold specificity sensitivity accuracy precision
## threshold 8.117251e-06    0.952381    0.6666667 0.9444444 0.2857143

```

Random effect:CHE

```
#GLMM model random effect:CHE
library(MASS)
set.seed(4850)
suppressMessages(modpql2 <- glmmPQL(binary
~Sex+ALB+Age+ALT+ALT+BIL+AST+CHOL+CREA+GGT+PROT,
                        data=train, random= ~ 1|CHE, family=binomial))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

suppressMessages(drop1(modpql2, test="Chi"))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Single term deletions
##
## Model:
## binary ~ Sex + ALB + Age + ALT + ALT + BIL + AST + CHOL + CREA +
##          GGT + PROT
##          Df AIC LRT Pr(>Chi)
## <none>
## Sex      1
## ALB      1
## Age      1
## ALT      1
## BIL      1
## AST      1
## CHOL     1
## CREA     1
## GGT      1
## PROT     1

testm2 <- mutate(test,
                  glmm_pred = predict(modpql2,newdata = test,
```



```

## binary ~ CHE + ALB + Age + ALT + ALT + BIL + AST + CHOL + CREA +
##      GGT + PROT
##      Df AIC LRT Pr(>Chi)
## <none>
## CHE      1
## ALB      1
## Age      1
## ALT      1
## BIL      1
## AST      1
## CHOL     1
## CREA     1
## GGT      1
## PROT     1

testm3 <- mutate(test,
                  glmm_pred = predict(modpql3, newdata = test,
                                     type="response"))

suppressMessages(library(pROC))

roc3 <- roc(testm3$binary, testm3$glmm_pred)

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

coords(roc3, "best", ret=c("threshold",
                           "specificity", "sensitivity", "accuracy", "precision"))

##      threshold specificity sensitivity accuracy precision
## threshold 0.4924259    0.9937888      0.9375 0.9887006    0.9375

```

Deep learning

binary

```

library(dplyr)
library(keras)
library(tensorflow)
library(tfdatasets)

set.seed(4850)
# Store the overall correlation in `M`
M <- cor(train[,1:10])

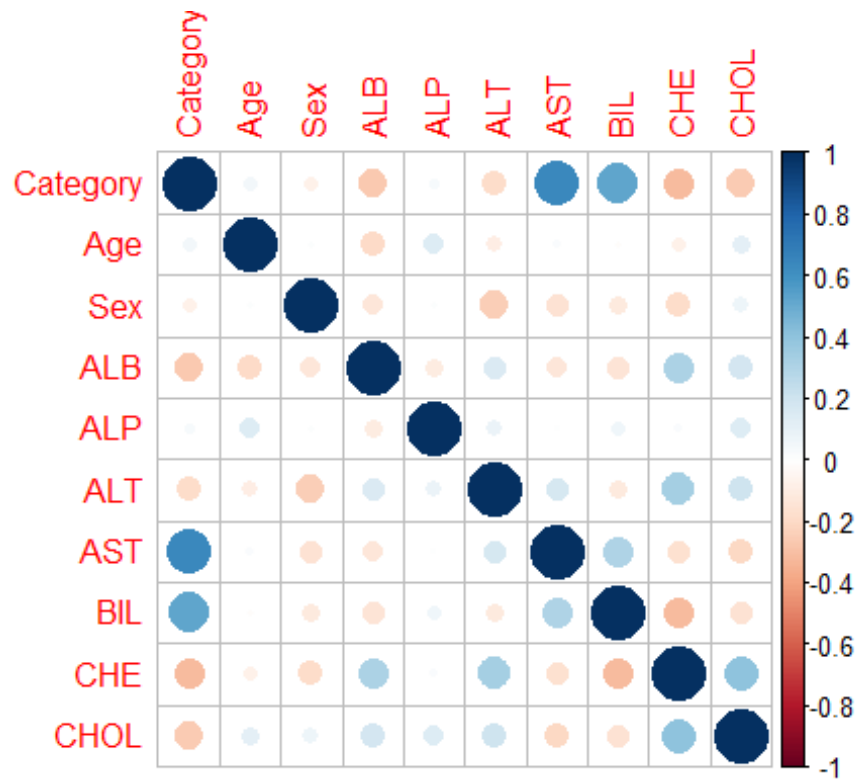
# Plot the correlation plot with `M`
library(corrplot)

```



```
## corrplot 0.84 loaded
```

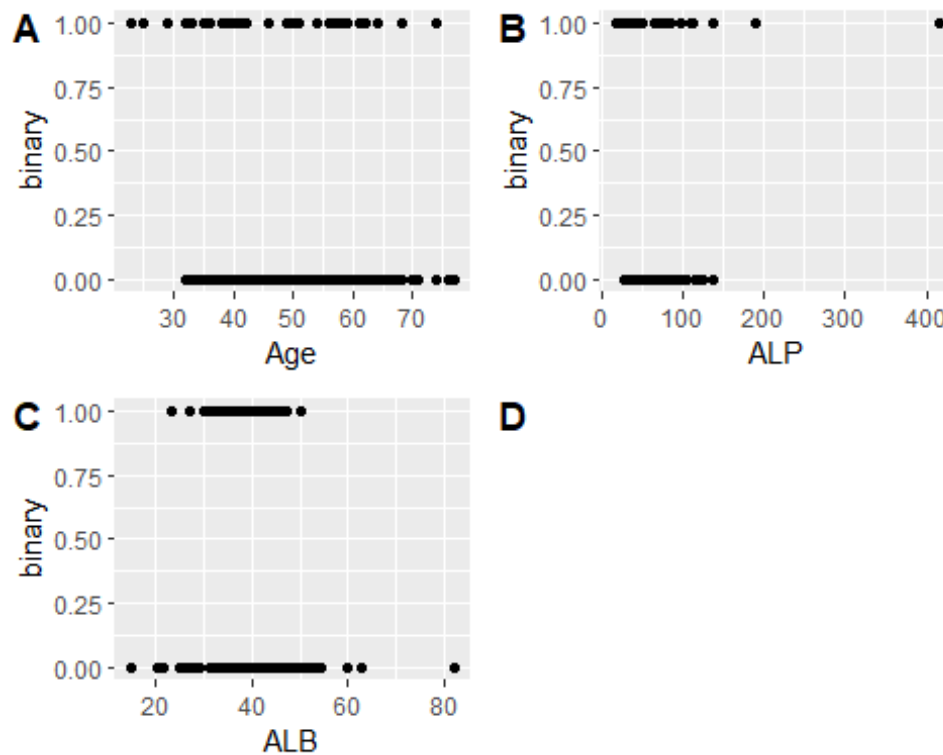
```
predictorsCorr=corrplot(M, method="circle")
```



```
figure <- ggarrange(agePlot, ALPPlot, ALBPlot, predictorsCorr,  
  labels = c("A", "B", "C", "D"),  
  ncol = 2, nrow = 2)
```

```
## Warning in as_grob.default(plot): Cannot convert object of class  
matrixarray  
## into a grob.
```

```
figure
```



```
#normalize
spec <- feature_spec(train, binary ~ .-Category ) %>%
  step_numeric_column(all_numeric(), normalizer_fn = scaler_standard()) %>%
  fit()

## Warning in normalizePath(path.expand(path), winslash, mustWork):
path[1]="C:
## \Users\Admin\.conda\envs\test-env/python.exe": The system cannot find the
file
## specified

## Warning in normalizePath(path.expand(path), winslash, mustWork):
path[1]="C:
## \Users\Admin\.conda\envs\test-env/python.exe": The system cannot find the
file
## specified

spec

## -- Feature Spec -----
-----
## A feature_spec with 12 steps.
## Fitted: TRUE
## -- Steps -----
-----
## The feature_spec has 1 dense features.
## StepNumericColumn: Age, Sex, ALB, ALP, ALT, AST, BIL, CHE, CHOL, CREA,
```

```

GGT, PROT
## -- Dense features -----
-----

layer <- layer_dense_features(
  feature_columns = dense_features(spec),
  dtype = tf$float32
)
suppressMessages(layer(train))

## tf.Tensor(
## [[ 0.653347 -0.6670947 -0.568868 ... -0.4312038 0.7929118
##      -0.76765865]
## [-0.44780117 -0.22397378 -0.13247223 ... -0.32766175 -0.35917082
##      -0.76765865]
## [-0.05207561 2.561358 0.41447717 ... 1.2864846 0.079717
##      1.2995006 ]
## ...
## [-0.9811694 -0.68571323 -1.2147336 ... 0.36570016 0.55518115
##      -0.76765865]
## [-1.1188128 0.05530416 -0.1441094 ... -0.14276527 -2.059866
##      1.2995006 ]
## [ 0.08556774 -0.71922666 0.8683287 ... 0.6338001 -0.03000497
##      -0.76765865]], shape=(412, 12), dtype=float32)

input <- layer_input_from_dataset(train[,2:13])

output <- input %>%
  layer_dense_features(dense_features(spec)) %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dense(units = 1)

dpmod <- keras_model(input, output)

dpmod %>%
  compile(
    loss = "mse",
    optimizer = optimizer_rmsprop(),
    metrics = list("mean_absolute_error")
  )

build_model <- function() {
  input <- layer_input_from_dataset(train[,2:13])

```

```

output <- input %>%
  layer_dense_features(dense_features(spec)) %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dense(units = 1)

dpmod <- keras_model(input, output)

dpmod %>%
  compile(
    loss = "mse",
    optimizer = optimizer_rmsprop(),
    metrics = list("mean_absolute_error")
  )

dpmod
}

# Display training progress by printing a single dot for each completed
epoch.
print_dot_callback <- callback_lambda(
  on_epoch_end = function(epoch, logs) {
    if (epoch %% 80 == 0) cat("\n")
    cat(".")
  }
)

dpmod <- build_model()

history <- dpmod %>% fit(
  x = train[2:13],
  y = train$binary,
  epochs = 500,
  validation_split = 0.2,
  verbose = 0,
  callbacks = list(print_dot_callback)
)

##
##
.....
...
##
.....
...
##
.....
...

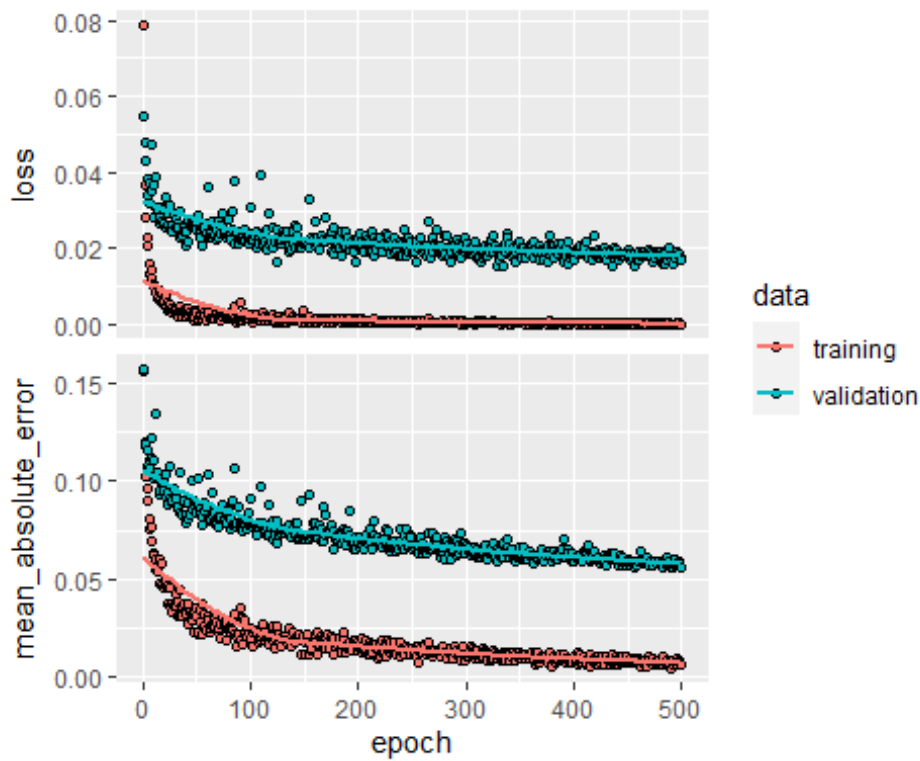
```

```
##
.....
...
##
.....
...
##
.....
...
## .....

library(ggplot2)

plot(history)

## `geom_smooth()` using formula 'y ~ x'
```



```
# it will stop when no more improvement
early_stop <- callback_early_stopping(monitor = "val_loss", patience = 20)

dpmod <- build_model()

history <- dpmod %>% fit(
  x = train[2:13],
  y = train$binary,
  epochs = 500,
  validation_split = 0.2,
```

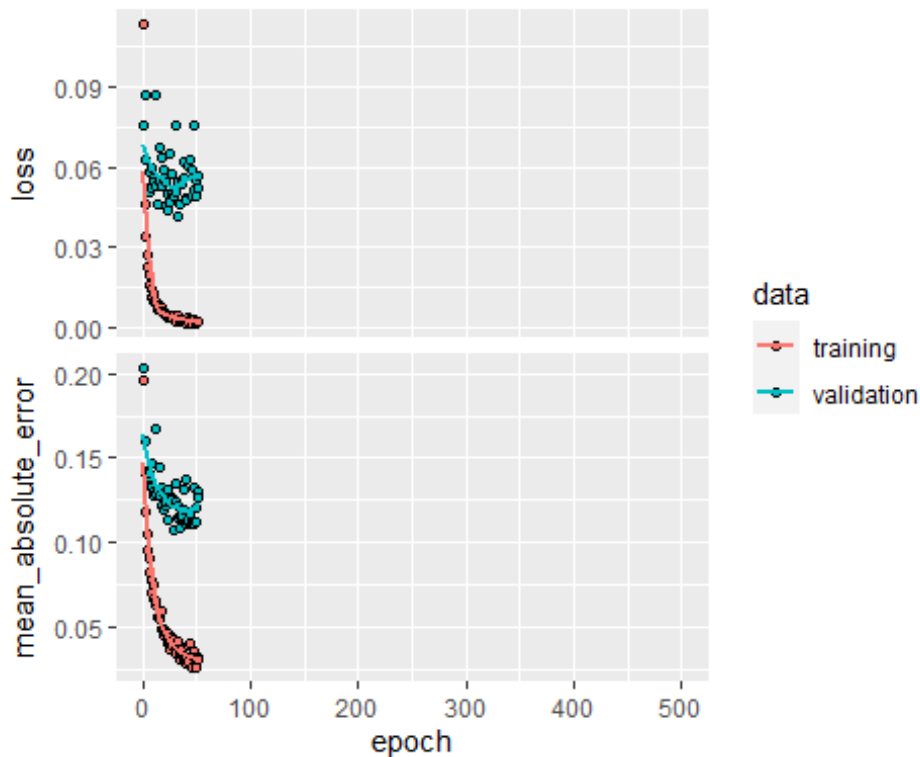
```

    verbose = 0,
    callbacks = list(early_stop)
)

plot(history)

## `geom_smooth()` using formula 'y ~ x'

```



```

suppressMessages(library(dplyr))
testm <- mutate(test,
                 dp_pred = dpmod %>% predict(test[,2:13]), type="response")

rocn <- roc(testm$binary, testm$dp_pred)

## Setting levels: control = 0, case = 1

## Warning in roc.default(testm$binary, testm$dp_pred): Deprecated use a
matrix as
## predictor. Unexpected results may be produced, please pass a numeric
vector.

## Setting direction: controls < cases

```

```
AUCnn <- auc(rocnn)
rs <- rocnn[['rocs']]
suppressMessages(summary(dpmmod))
```

```
## Model: "model_2"
##
```

## Layer (type)	Output Shape	Param #	Connected to
-----------------	--------------	---------	--------------

=====

===

## ALB (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## ALP (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## ALT (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## AST (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## Age (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## BIL (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## CHE (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## CHOL (InputLayer)	[(None,)]	0	
----------------------	-----------	---	--

## CREA (InputLayer)	[(None,)]	0	
----------------------	-----------	---	--

## GGT (InputLayer)	[(None,)]	0	
---------------------	-----------	---	--

## PROT (InputLayer)	[(None,)]	0
##		

## Sex (InputLayer)	[(None,)]	0
##		

## dense_features_3 (DenseFe (None, 12)	0	ALB[0][0]
##		ALP[0][0]
##		ALT[0][0]
##		AST[0][0]
##		Age[0][0]
##		BIL[0][0]
##		CHE[0][0]
##		CHOL[0][0]
##		CREA[0][0]
##		GGT[0][0]
##		PROT[0][0]
##		Sex[0][0]

## dense_8 (Dense)	(None, 64)	832
dense_features_3[0][0]		
##		

## dense_7 (Dense)	(None, 64)	4160	dense_8[0][0]
##			

## dense_6 (Dense)	(None, 1)	65	dense_7[0][0]
##			

=====

===

Total params: 5,057

Trainable params: 5,057

Non-trainable params: 0

##

##RM,(AST ALT ALP) glmm, nn

coords(rocrf,"best",**ret=c**("threshold",

"specificity", "sensitivity", "accuracy","precision"))*##the random forest*

model


```

##          threshold specificity sensitivity accuracy precision
## threshold 0.2833333    0.9689441          1 0.9717514 0.7619048

coords(roc1,"best",ret=c("threshold",
"specificity", "sensitivity", "accuracy","precision"))# the AST GLMM model

##          threshold specificity sensitivity accuracy precision
## threshold 8.117251e-06    0.952381    0.6666667 0.9444444 0.2857143

coords(roc2,"best",ret=c("threshold",
"specificity", "sensitivity", "accuracy","precision"))#the CHE GLMM model

##          threshold specificity sensitivity accuracy precision
## threshold 0.3358878          1    0.8571429 0.9863014          1

coords(roc3,"best",ret=c("threshold",
"specificity", "sensitivity", "accuracy","precision"))# the Sex GLMM model

##          threshold specificity sensitivity accuracy precision
## threshold 0.4924259    0.9937888          0.9375 0.9887006    0.9375

coords(rocnn,"best",ret=c("threshold",
"specificity", "sensitivity", "accuracy","precision"))# the nn GLMM model

##          threshold specificity sensitivity accuracy precision
## threshold 0.2531132    0.9813665          1 0.9830508 0.8421053

#importance: impurity gini importance
c(importance(rfmod),importance(rfmod))

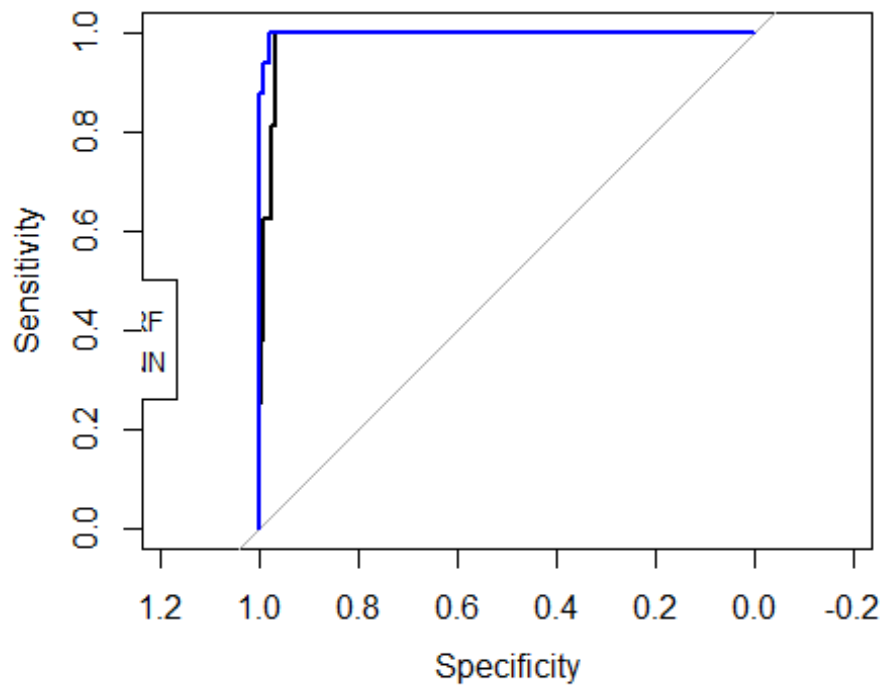
## [1] 3.610843e-01 2.775558e-18 4.248468e-01 3.345247e+00 7.598513e+00
## [6] 1.973172e+01 1.716077e+00 1.254098e+00 3.677537e-01 4.640634e-01
## [11] 2.672794e+00 6.762646e-15 3.610843e-01 2.775558e-18 4.248468e-01
## [16] 3.345247e+00 7.598513e+00 1.973172e+01 1.716077e+00 1.254098e+00
## [21] 3.677537e-01 4.640634e-01 2.672794e+00 6.762646e-15

log(importance(rfmod))

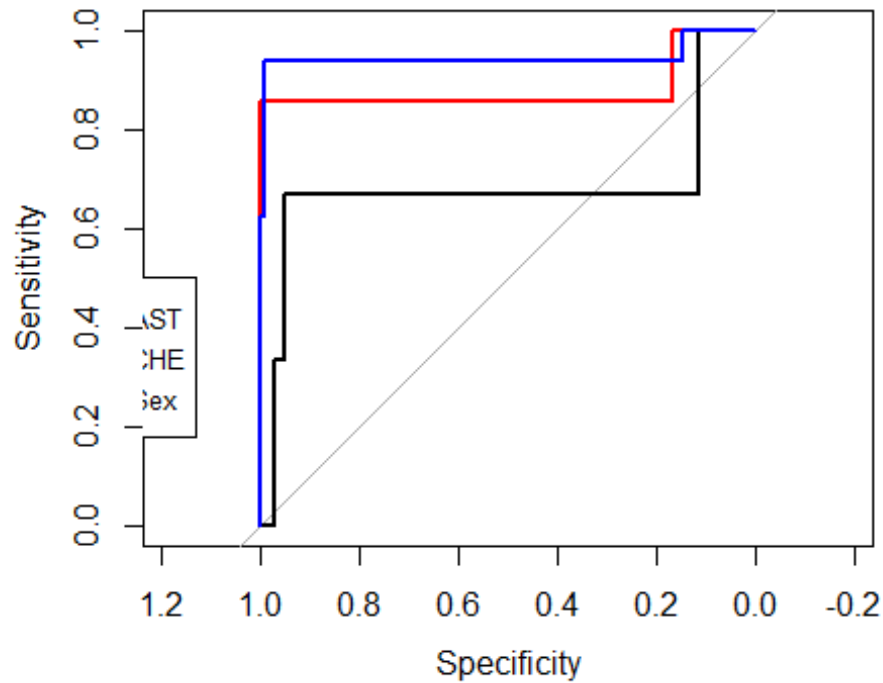
##          IncNodePurity
## Age          -1.0186438
## Sex          -40.4256800
## ALB          -0.8560265
## ALP           1.2075404
## ALT           2.0279526
## AST           2.9822275
## BIL           0.5400411
## CHE           0.2264169
## CHOL         -1.0003418
## CREA         -0.7677341
## GGT           0.9831243
## PROT        -32.6273622

```

```
#neural network and random forest
plot(rocrf)
plot(rocn, add=TRUE, col='blue')
legend(1.45, 0.5, legend=c("RF", "NN"),
      col=c("black", "blue"), lty=1, cex=0.8)
```



```
#three glmm roc curve
plot(roc1)
plot(roc2, add=TRUE, col='red')
plot(roc3, add=TRUE, col='blue')
legend(1.45, 0.5, legend=c("AST", "CHE", "Sex"),
      col=c("black", "red", "blue"), lty=1, cex=0.8)
```



```
#randomforest best glmm and nn roc curve
plot(rocrf)
plot(roc3, add=TRUE, col='red')
plot(rocnn, add=TRUE, col='blue')
legend(1.45, 0.5, legend=c("RF", "Sex-GLMM", "NN"),
      col=c("black", "red", "blue"), lty=1, cex=0.8)
```

