

A decorative graphic on the left side of the slide, consisting of a network of white lines and small circles on a blue gradient background, resembling a circuit board or a neural network.

# TODD GARNER

DS 6306 WEEK 7 PART 2

FEBRUARY 14. 2023

# PART 2, #1

- Read in the data, add sex, run code as below with seed(4), view results.

```
##{r}
# Read in the training set. Check to "view" the full file to make sure it's what we want.
Titanic <- read.csv(file.choose(), header = TRUE)
view(Titanic)
##

We won't need to run that piece of code again so I'm isolating it.

##{r}
#Titanic$SurvivedF <- factor(Titanic$Survived, labels = c("Died", "Survived"))
Titanic$MF <- factor(Titanic$Sex, labels = c("0", "1")) #male = 1, female = 0
head(Titanic$MF)
head(Titanic)
Titanic_sub <- Titanic %>% filter(!is.na(Age) & !is.na(Pclass) & !is.na(MF))
Titanic_sub_filter <- Titanic_sub %>% select(Age, Pclass, Survived, MF)
head(Titanic_sub_filter)
model <- naiveBayes(Titanic_sub_filter[[(Titanic_sub_filter$Age) & (Titanic_sub_filter$Pclass) & (Titanic_sub_filter$MF)], c("Age", "Pclass",
"MF")], Titanic_sub_filter$Survived, laplace = 1)
Titanic_clean = Titanic %>% filter(!is.na(Age) & !is.na(Pclass) & !is.na(MF))
set.seed(4)
trainIndices = sample(seq(1:length(Titanic_clean$MF)),round(.7*length(Titanic_clean$MF)))
trainTitanic = Titanic_clean[trainIndices,]
testTitanic = Titanic_clean[-trainIndices,]
head(trainTitanic)
dim(trainTitanic)
head(testTitanic)
dim(testTitanic)
model <- naiveBayes(trainTitanic,as.factor(trainTitanic$Survived) , laplace = 1)
summary(model)
dim(model)
#head(model)
df <- data.frame(testTitanic)
nrow(df)
x <- round(predict(model, df, type = "raw"), digits = 0)
y <- x[,2]
y
dim(df$Survived)
nrow(df$Survived)
table(y, df$Survived)
confusionMatrix(table(y, df$Survived))
##
```

## Confusion Matrix and Statistics

y	0	1
0	128	0
1	0	86

Accuracy : 1  
95% CI : (0.9829, 1)  
No Information Rate : 0.5981  
P-Value [ACC > NIR] : < 2.2e-16

Kappa : 1

Mcnemar's Test P-Value : NA

Sensitivity : 1.0000  
Specificity : 1.0000  
Pos Pred Value : 1.0000  
Neg Pred Value : 1.0000  
Prevalence : 0.5981  
Detection Rate : 0.5981  
Detection Prevalence : 0.5981  
Balanced Accuracy : 1.0000

'Positive' Class : 0

- Perhaps I'm missing the point, but this looks identical to the prior exercise.

# P2 #2, WRITE LOOP(100) AND GATHER RESULTS

Code:

```
####{r}
Titanic_clean = Titanic %>% filter(!is.na(Age) & !is.na(Pclass) & !is.na(MF))
iterations = 100
master_sens <- 0
master_spec <- 0
master_acc <- 0
master_sens <- data.frame(master_sens)
master_spec <- data.frame(master_spec)
master_acc <- data.frame(master_acc)

for(i in 1:iterations) {
  set.seed(i)
  trainIndices = sample(seq(1:length(Titanic_clean$MF)),round(.7*length(Titanic_clean$MF)))
  trainTitanic = Titanic_clean[trainIndices,]
  testTitanic = Titanic_clean[-trainIndices,]

  model <- naiveBayes(trainTitanic,as.factor(trainTitanic$Survived) , laplace = 1)
  df <- data.frame(testTitanic)
  x <- round(predict(model, df, type = "raw"), digits = 0)
  y <- x[,2]
  y
  master_sens[,i] = sensitivity(factor(y), factor(df$Survived))
  master_spec[,i] = specificity(factor(y), factor(df$Survived))
  z <- table(factor(y), factor(df$Survived))
  CM <- confusionMatrix(z, k = i)
  master_acc[,i] = CM$overall[1]

}

mean_sens = colMeans(master_sens)
mean_spec = colMeans(master_spec)
mean_acc = colMeans(master_acc)

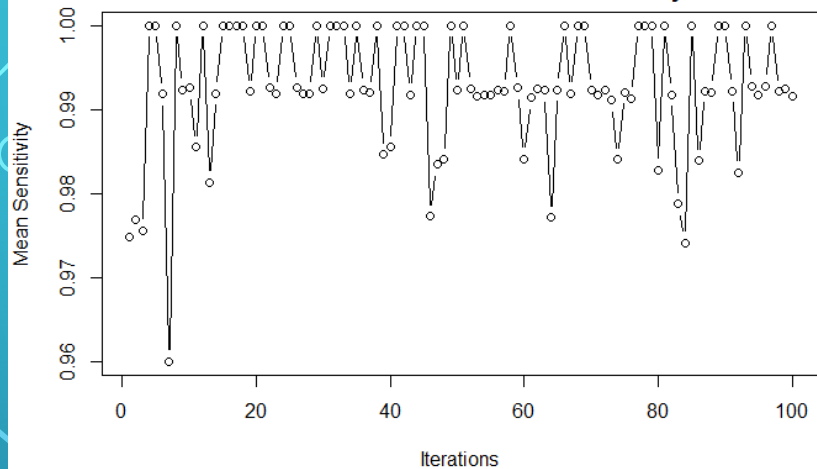
which.max(mean_sens)
max(mean_sens)
which.max(mean_spec)
max(mean_spec)
which.max(mean_acc)
max(mean_acc)

plot(mean_sens,xlab = "Iterations", ylab = "Mean Sensitivity", main = "Seed iterations versus Mean Sensitivity", type = "b")
plot(mean_spec,xlab = "Iterations", ylab = "Mean Specificity", main = "Seed iterations versus Mean Specificity",type = "b")
plot(mean_acc, xlab = "Iterations", ylab = "Mean Accuracy", main = "Seed iterations versus Mean Accuracy",type = "b")
####
```

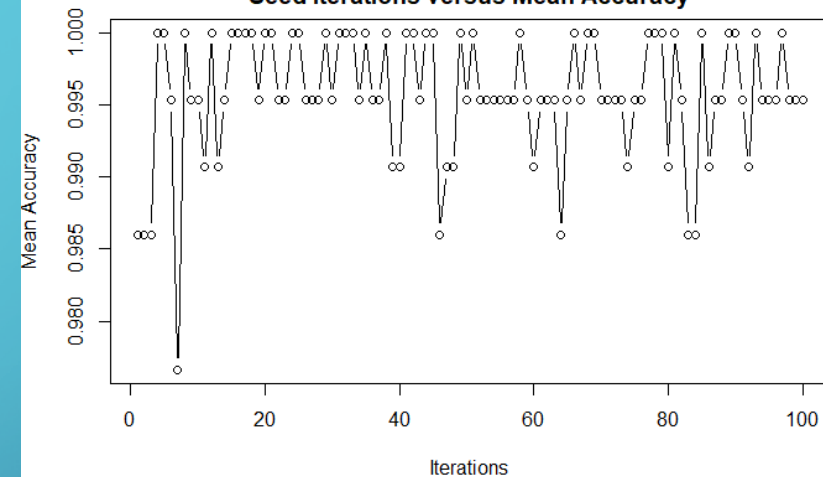
# P2 #3, PLOTS OF ACCURACY, SENSITIVITY, SPECIFICITY

- Means were all identical: 1

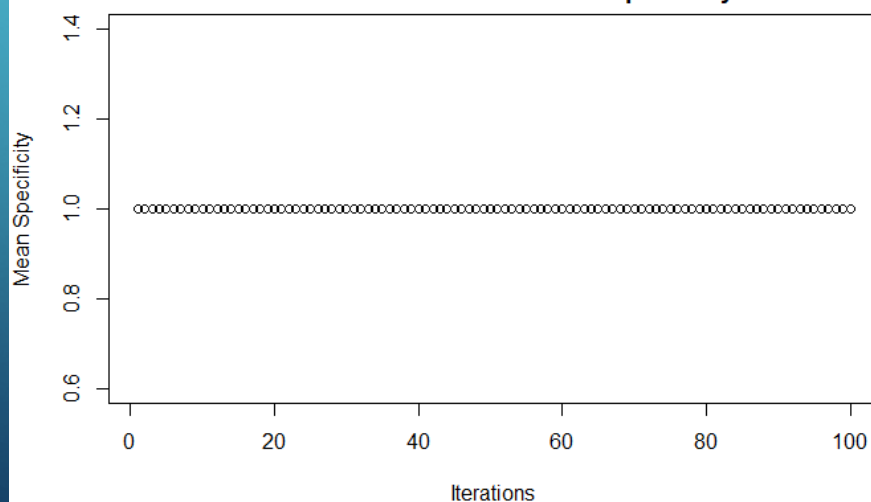
Seed iterations versus Mean Sensitivity



Seed iterations versus Mean Accuracy



Seed iterations versus Mean Specificity





# P2 IRIS, CROSS VALIDATION SEPAL LENGTH & WIDTH

I must say I was stumped by this one. My code is found below. Perhaps in Live Session we can go through it to find out what I'm doing wrong?

```
25 # Part 2, Iris
26
27 ```{r}
28 #View(iris)
29 iris_clean = iris %>% filter(!is.na(Sepal.Length) & !is.na(Sepal.Width))
30 iterations = 100
31
32 for(i in 1:iterations) {
33   set.seed(i)
34   trainiris = sample(seq(1:length(iris_clean$Sepal.Length)),round(.7*length(iris_clean$Sepal.Length)))
35   trainIris = iris_clean[trainiris,]
36   testIris = iris_clean[-trainiris,]
37   head(trainiris)
38   head(testIris)
39
40   model <- naiveBayes(trainIris,as.factor(trainIris$Sepal.Length & trainIris$Sepal.Width), laplace = 1) # & trainIris$Sepal.Width
41   head(model)
42   df <- data.frame(testIris)
43   head(df)
44   x <- round(predict(model, df, type = "raw"), digits = 1)
45   x
46   y <- x[,2]
47   y
48   master_sens[,i] = sensitivity(factor(y), factor(df$Sepal.Length))
49   master_spec[,i] = specificity(factor(y), factor(df$Sepal.Length))
50   CM <- confusionMatrix(factor(y), factor(df$Sepal.Length), k = i)
51   master_acc[,i] = CM$overall
52 }
53
54 mean_sens = colMeans(master_sens)
55 mean_spec = colMeans(master_spec)
56 mean_acc = colMeans(master_acc)
57
58 which.max(mean_sens)
59 max(mean_sens)
60 which.max(mean_spec)
61 max(mean_spec)
62 which.max(mean_acc)
63 max(mean_acc)
64
```