final\_rmd

### Problem 4  
### parity (16)  
parity\_n=16  
empty\_mat <- matrix(NA, nrow = 2^parity\_n, ncol = parity\_n)  
  
# count down inside count up outside  
inner\_count <- parity\_n; outer\_count <- 0  
for (i in 1:parity\_n) {  
 # print(paste0('i', inner\_count - i, 'o', outer\_count + i))  
 new\_vec <- rep(rep(c(1, 0), each = 2^(inner\_count - i)), 2^(outer\_count + i - 1))  
 empty\_mat[,i] <- new\_vec  
}  
  
empty\_mat\_df <- as.data.frame(empty\_mat)  
  
# XOR  
# (A AND B)  
a\_and\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 AND 2  
 a\_and\_b <- empty\_mat\_df[,i] & empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 AND 3,   
 a\_and\_b <- empty\_mat\_df[,i] & a\_and\_b  
 }  
}  
a\_and\_b <- a\_and\_b \* 1  
  
# (A OR B)  
a\_or\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 OR 2  
 a\_or\_b <- empty\_mat\_df[,i] | empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 OR 3,   
 a\_or\_b <- empty\_mat\_df[,i] | a\_or\_b  
 }  
}  
a\_or\_b <- a\_or\_b \* 1  
  
not\_a\_and\_b <- (!a\_and\_b) \* 1  
  
# (A OR B) OR NOT (A AND B)  
a\_or\_b\_or\_not\_a\_and\_b <- (a\_or\_b | not\_a\_and\_b) \* 1  
  
library(neuralnet)  
X <- empty\_mat\_df  
Y <- a\_or\_b\_or\_not\_a\_and\_b  
df <- cbind(Y, X)  
net1 <- neuralnet(Y~., df, hidden=2)  
plot(net1)  
  
y <- predict(net1, X)  
sum(round(y) == Y)  
  
### parity(127)  
parity\_n <- 127  
empty\_mat <- matrix(0, nrow = 2^parity\_n, ncol = parity\_n)  
  
# count down inside count up outside  
inner\_count <- parity\_n; outer\_count <- 0  
for (i in 1:parity\_n) {  
 # print(paste0('i', inner\_count - i, 'o', outer\_count + i))  
 new\_vec <- rep(rep(c(1, 0), each = 2^(inner\_count - i)), 2^(outer\_count + i - 1))  
 empty\_mat[,i] <- new\_vec  
}  
  
empty\_mat\_df <- as.data.frame(empty\_mat)  
write.csv(empty\_mat\_df, file = 'truth\_table127.csv', row.names = FALSE)  
  
# XOR  
# (A AND B)  
a\_and\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 AND 2  
 a\_and\_b <- empty\_mat\_df[,i] & empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 AND 3,   
 a\_and\_b <- empty\_mat\_df[,i] & a\_and\_b  
 }  
}  
a\_and\_b <- a\_and\_b \* 1  
  
# (A OR B)  
a\_or\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 OR 2  
 a\_or\_b <- empty\_mat\_df[,i] | empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 OR 3,   
 a\_or\_b <- empty\_mat\_df[,i] | a\_or\_b  
 }  
}  
a\_or\_b <- a\_or\_b \* 1  
  
not\_a\_and\_b <- (!a\_and\_b) \* 1  
  
# (A OR B) OR NOT (A AND B)  
a\_or\_b\_or\_not\_a\_and\_b <- (a\_or\_b | not\_a\_and\_b) \* 1  
  
### parity (8)  
parity\_n=8  
empty\_mat <- matrix(NA, nrow = 2^parity\_n, ncol = parity\_n)  
  
# count down inside count up outside  
inner\_count <- parity\_n; outer\_count <- 0  
for (i in 1:parity\_n) {  
 # print(paste0('i', inner\_count - i, 'o', outer\_count + i))  
 new\_vec <- rep(rep(c(1, 0), each = 2^(inner\_count - i)), 2^(outer\_count + i - 1))  
 empty\_mat[,i] <- new\_vec  
}  
  
empty\_mat\_df <- as.data.frame(empty\_mat)  
  
# XOR  
# (A AND B)  
a\_and\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 AND 2  
 a\_and\_b <- empty\_mat\_df[,i] & empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 AND 3,   
 a\_and\_b <- empty\_mat\_df[,i] & a\_and\_b  
 }  
}  
a\_and\_b <- a\_and\_b \* 1  
  
# (A OR B)  
a\_or\_b <- rep(NA, 2^parity\_n)  
for (i in 1:(parity\_n - 1)) {  
 if (i == 1) {  
 # 1 OR 2  
 a\_or\_b <- empty\_mat\_df[,i] | empty\_mat\_df[,i+1]   
 } else if ((i != 1) & (i != 2)) {  
 # 2 OR 3,   
 a\_or\_b <- empty\_mat\_df[,i] | a\_or\_b  
 }  
}  
a\_or\_b <- a\_or\_b \* 1  
  
not\_a\_and\_b <- (!a\_and\_b) \* 1  
  
# (A OR B) OR NOT (A AND B)  
a\_or\_b\_or\_not\_a\_and\_b <- (a\_or\_b | not\_a\_and\_b) \* 1  
  
X <- empty\_mat\_df  
Y <- a\_or\_b\_or\_not\_a\_and\_b  
df <- cbind(Y, X)  
net3 <- neuralnet(Y~., df, hidden=2)  
plot(net3)  
y <- predict(net3, X)  
sum(round(y) == Y)  
2^8  
  
part\_a <- c(1,0,1,0,1,0,1,0)  
part\_a <- data.frame(t(data.frame(part\_a)))  
colnames(part\_a) <- colnames(X)  
predict(net3, part\_a)  
  
part\_b <- c(1,1,0,0,0,1,1,0)  
part\_b <- data.frame(t(data.frame(part\_b)))  
colnames(part\_b) <- colnames(X)  
predict(net3, part\_b)  
  
part\_c <- c(1,0,0,0,1,0,0,0)  
part\_c <- data.frame(t(data.frame(part\_c)))  
colnames(part\_c) <- colnames(X)  
predict(net3, part\_c)  
  
part\_d <- c(1,1,1,1,1,1,1,1)  
part\_d <- data.frame(t(data.frame(part\_d)))  
colnames(part\_d) <- colnames(X)  
predict(net3, part\_d)  
  
### Problem 5  
library(mclust)  
N <- 500  
X1 <- rnorm(n = N, mean = 1, sd = sqrt(0.1))  
X2 <- rnorm(n = N, mean = 2, sd = sqrt(0.1))  
X3 <- rnorm(n = N, mean = 3, sd = sqrt(0.2))  
P1 <- 1/6  
P2 <- 1/2  
P3 <- 1/3  
  
X\_mix <- (X1 \* P1) + (X2 \* P2) + (X3 \* P3)  
em\_clustering = Mclust(data = X\_mix, G = 3)  
em\_clustering$parameters