Eyewitness identification data analysis

Kym McCormick

15 May 2019

din <- data %>%  
   
 ## Select required variables ---  
 select(uid,  
 condition,   
 Test\_T1\_suspectIdentified,   
 confidence\_rating,   
 demographics\_age,   
 demographics\_gender,   
 demographics\_country)%>%  
   
 ## Only include data from USA and includes a confidence rating ----  
 filter(demographics\_country == "USA", !is.na(confidence\_rating)) %>%  
   
 ## Separate the confounded condition variable into separate variables ----  
 separate(condition,c("memory","expectation","target")) %>%  
   
 ## Create boolean variables for each identification outcome ---  
 mutate(CID = if\_else(target == "P" & Test\_T1\_suspectIdentified == "F68", 1, 0),  
 Miss = if\_else(target == "P" & Test\_T1\_suspectIdentified == "Silhouette", 1, 0),  
 TPFoilID = if\_else(target == "P" & Test\_T1\_suspectIdentified != "F68" & Test\_T1\_suspectIdentified != "Silhouette", 1, 0),  
 CR = if\_else(target == "A" & Test\_T1\_suspectIdentified == "Silhouette", 1, 0),  
 TAFoilID = if\_else(target == "A" & Test\_T1\_suspectIdentified != "Silhouette", 1, 0)  
 )  
  
## There are twice the number of target present lineups than target absent lineups. To rebalance this in a way that preserves the groupings of confidence intervals, I have chosen to double the Target Absent data before grouping into equal sized confidence rating bins. The following function does this and returns the data matrix with a new variable: decile.  
  
Con\_Perc <- function(data) {  
 ## Separate data into target present and absent dataframes ---  
 Ratings\_Absent <- data %>%   
 filter(target == "A")  
   
 Ratings\_Present <- data %>%  
 filter(target == "P")  
   
 ## Bind back together, doubling up on the target absent group ---  
 ## NOTE THAT DUPLICATES MAY BE REMOVED USING THE UID NUMBERS ---  
 Ratings <- rbind(Ratings\_Absent,Ratings\_Absent,Ratings\_Present)  
   
 ## Group chooser confidence into decile confidence rating bins  
 din\_mod\_choose <- Ratings %>%  
 filter(Test\_T1\_suspectIdentified != "Silhouette") %>%  
 mutate(decile\_cr = ntile(confidence\_rating,10)  
 )  
   
 ## Group non-chooser confidence into a single zero confidence rating bin  
 din\_mod\_nochoose <- Ratings %>%  
 filter(Test\_T1\_suspectIdentified == "Silhouette") %>%  
 mutate(decile\_cr = 0)  
   
 ## Bind chooser and non-chooser data back together  
 din\_mod <- rbind(din\_mod\_choose, din\_mod\_nochoose)  
 return(din\_mod)  
}  
  
## Apply the above function to the cleaned dataset  
din\_mod <- Con\_Perc(din)  
  
## Check the grouping of the confidence rating deciles to ensure evenness  
din\_mod %>%  
 select(decile\_cr) %>%  
 map(~prop.table(table(.)))

## $decile\_cr  
## .  
## 0 1 2 3 4 5   
## 0.24442765 0.07555723 0.07555723 0.07555723 0.07555723 0.07555723   
## 6 7 8 9 10   
## 0.07555723 0.07555723 0.07555723 0.07555723 0.07555723

## Demographics

age <- describe(din$demographics\_age)  
age

## vars n mean sd median trimmed mad min max range skew kurtosis  
## X1 1 2012 37.15 11.42 35 35.91 10.38 18 81 63 0.92 0.26  
## se  
## X1 0.25

gender <- din %>%   
 select(demographics\_gender) %>%   
 map(~prop.table(table(.)))  
gender

## $demographics\_gender  
## .  
## female male other   
## 0.000000000 0.509443340 0.484095427 0.006461233

## Identification counts

## Counts within each confidence level and accross manipulation groups ---  
ID\_count <- din\_mod %>%   
 group\_by(decile\_cr,memory,expectation) %>%  
 summarise(  
 Miss = sum(Miss),  
 TPFID = sum(TPFoilID),  
 Correct\_ID = sum(CID),  
 CR = sum(CR),  
 TAFID = sum(TAFoilID)\*7/8,  
 False\_ID = sum(TAFoilID)/8,  
 TP = sum(Miss,TPFoilID,CID),  
 TA = sum(CR,TAFID,False\_ID)  
 ) %>%  
 ungroup()  
  
head(ID\_count,10)

## # A tibble: 10 x 11  
## decile\_cr memory expectation Miss TPFID Correct\_ID CR TAFID False\_ID  
## <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0 S H 35 0 0 94 0 0   
## 2 0 S L 74 0 0 104 0 0   
## 3 0 W H 64 0 0 70 0 0   
## 4 0 W L 88 0 0 118 0 0   
## 5 1 S H 0 15 6 0 26.2 3.75  
## 6 1 S L 0 14 5 0 14.9 2.12  
## 7 1 W H 0 17 9 0 28 4   
## 8 1 W L 0 14 6 0 30.6 4.38  
## 9 2 S H 0 12 17 0 27.1 3.88  
## 10 2 S L 0 14 9 0 21 3   
## # ... with 2 more variables: TP <dbl>, TA <dbl>

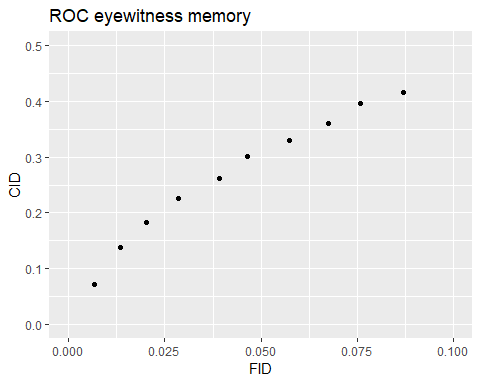
## ROC data

## Creating ROC data   
  
  
ROC <- ID\_count %>%  
 ## Group by binned confidence level groups ---  
 group\_by(decile\_cr) %>%  
   
 ## Find sums for correct identifications, false identifications, and the size of the target present and target absent groups ---   
 summarise(  
 Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TA = sum(TA),  
 TP = sum(TP)  
 )   
  
## Reverse the order of the confidence level groups (from highest to lowest) ---  
ROC <- ROC[11:1,]  
  
## Calculate the cummulative counts ---  
ROC <- ROC %>%  
 mutate(  
 Correct\_ID\_cum = cumsum(Correct\_ID),  
 False\_ID\_cum = cumsum(False\_ID)  
 )%>%  
 ## Calclate the cummulative CID and FID proportions ---   
 ungroup()%>%  
 mutate(  
 CID = Correct\_ID\_cum/sum(TP),  
 FID = False\_ID\_cum/sum(TA)  
 )  
  
ROC

## # A tibble: 11 x 9  
## decile\_cr Correct\_ID False\_ID TA TP Correct\_ID\_cum False\_ID\_cum  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 10 100 8.5 68 132 100 8.5  
## 2 9 91 8.62 69 131 191 17.1  
## 3 8 62 8.5 68 132 253 25.6  
## 4 7 58 10.5 84 116 311 36.1  
## 5 6 49 13.5 108 92 360 49.6  
## 6 5 55 9.38 75 125 415 59   
## 7 4 38 14 112 88 453 73   
## 8 3 43 12.8 102 98 496 85.8  
## 9 2 50 10.5 84 116 546 96.2  
## 10 1 26 14.2 114 86 572 110.   
## 11 0 0 0 386 261 572 110.   
## # ... with 2 more variables: CID <dbl>, FID <dbl>

### Plotting ROC

library(ggplot2)  
# Basic scatter plot  
ggplot(ROC, aes(x=FID, y=CID)) +   
 geom\_point() +  
 xlim(0,.1) +  
 ylim(0,.5) +  
 ggtitle("ROC eyewitness memory")

 ##ROC by levels of memory

## Creating ROC data for each manipulation group   
  
  
ROC\_memory <- ID\_count %>%  
 ## Group by binned confidence level groups ---  
 group\_by(decile\_cr, memory) %>%  
   
 ## Find sums for correct identifications, false identifications, and the size of the target present and target absent groups ---   
 summarise(  
 Correct\_ID = sum(Correct\_ID),  
 False\_ID = sum(False\_ID),  
 TA = sum(TA),  
 TP = sum(TP)  
 ) %>%  
 ungroup()  
  
## Reverse the order of the confidence level groups (from highest to lowest) ---  
ROC\_memory <- ROC\_memory[22:1,]  
head(ROC\_memory,10)

## # A tibble: 10 x 6  
## decile\_cr memory Correct\_ID False\_ID TA TP  
## <dbl> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 10 W 52 3.75 30 65  
## 2 10 S 48 4.75 38 67  
## 3 9 W 30 5.12 41 46  
## 4 9 S 61 3.5 28 85  
## 5 8 W 27 3.75 30 62  
## 6 8 S 35 4.75 38 70  
## 7 7 W 19 4.75 38 45  
## 8 7 S 39 5.75 46 71  
## 9 6 W 19 6.62 53 44  
## 10 6 S 30 6.88 55 48

## Calculate the cummulative counts for weak memory---  
ROC\_W <- ROC\_memory %>%  
 filter(memory == "W") %>%  
 mutate(  
 Correct\_ID\_cum = cumsum(Correct\_ID),  
 False\_ID\_cum = cumsum(False\_ID)  
 ) %>%  
 ## Calclate the cummulative CID and FID proportions ---   
 mutate(  
 CID = Correct\_ID\_cum/sum(TP),  
 FID = False\_ID\_cum/sum(TA)  
 )  
ROC\_W

## # A tibble: 11 x 10  
## decile\_cr memory Correct\_ID False\_ID TA TP Correct\_ID\_cum  
## <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 10 W 52 3.75 30 65 52  
## 2 9 W 30 5.12 41 46 82  
## 3 8 W 27 3.75 30 62 109  
## 4 7 W 19 4.75 38 45 128  
## 5 6 W 19 6.62 53 44 147  
## 6 5 W 23 4.75 38 53 170  
## 7 4 W 14 6 48 37 184  
## 8 3 W 13 5.75 46 38 197  
## 9 2 W 24 3.62 29 64 221  
## 10 1 W 15 8.38 67 46 236  
## 11 0 W 0 0 188 152 236  
## # ... with 3 more variables: False\_ID\_cum <dbl>, CID <dbl>, FID <dbl>

## Calculate the cummulative counts for strong memory---  
ROC\_S <- ROC\_memory %>%  
 filter(memory == "S") %>%  
 mutate(  
 Correct\_ID\_cum = cumsum(Correct\_ID),  
 False\_ID\_cum = cumsum(False\_ID)  
 ) %>%  
 ## Calclate the cummulative CID and FID proportions ---   
 mutate(  
 CID = Correct\_ID\_cum/sum(TP),  
 FID = False\_ID\_cum/sum(TA)  
 )  
  
ROC\_S

## # A tibble: 11 x 10  
## decile\_cr memory Correct\_ID False\_ID TA TP Correct\_ID\_cum  
## <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 10 S 48 4.75 38 67 48  
## 2 9 S 61 3.5 28 85 109  
## 3 8 S 35 4.75 38 70 144  
## 4 7 S 39 5.75 46 71 183  
## 5 6 S 30 6.88 55 48 213  
## 6 5 S 32 4.62 37 72 245  
## 7 4 S 24 8 64 51 269  
## 8 3 S 30 7 56 60 299  
## 9 2 S 26 6.88 55 52 325  
## 10 1 S 11 5.88 47 40 336  
## 11 0 S 0 0 198 109 336  
## # ... with 3 more variables: False\_ID\_cum <dbl>, CID <dbl>, FID <dbl>

combined <- ROC%>%  
 filter(decile\_cr != "0")%>%  
 select(CID,FID)%>%  
 mutate(  
 nCID =1-CID,  
 nFID = 1-FID  
 )%>%  
 select(CID,nCID,FID,nFID)%>%  
 t()%>%  
 as.vector()  
combined

## [1] 0.072621641 0.927378359 0.006692913 0.993307087 0.138707335  
## [6] 0.861292665 0.013484252 0.986515748 0.183732752 0.816267248  
## [11] 0.020177165 0.979822835 0.225853304 0.774146696 0.028444882  
## [16] 0.971555118 0.261437908 0.738562092 0.039074803 0.960925197  
## [21] 0.301379811 0.698620189 0.046456693 0.953543307 0.328976035  
## [26] 0.671023965 0.057480315 0.942519685 0.360203341 0.639796659  
## [31] 0.067519685 0.932480315 0.396514161 0.603485839 0.075787402  
## [36] 0.924212598 0.415395788 0.584604212 0.087007874 0.912992126

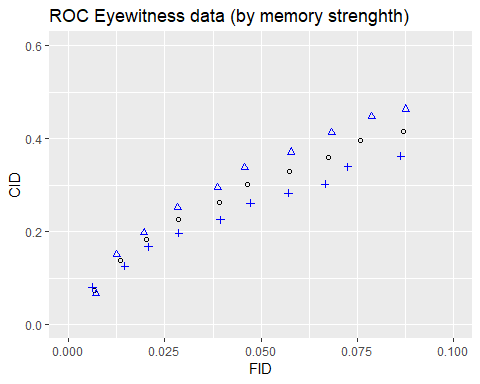
strong <- ROC\_S %>%  
 filter(decile\_cr != "0")%>%  
 select(CID,FID)%>%  
 mutate(  
 nCID =1-CID,  
 nFID = 1-FID  
 )%>%  
 select(CID,nCID,FID,nFID)%>%t()%>%  
 as.vector()  
strong

## [1] 0.066206897 0.933793103 0.007175227 0.992824773 0.150344828  
## [6] 0.849655172 0.012462236 0.987537764 0.198620690 0.801379310  
## [11] 0.019637462 0.980362538 0.252413793 0.747586207 0.028323263  
## [16] 0.971676737 0.293793103 0.706206897 0.038708459 0.961291541  
## [21] 0.337931034 0.662068966 0.045694864 0.954305136 0.371034483  
## [26] 0.628965517 0.057779456 0.942220544 0.412413793 0.587586207  
## [31] 0.068353474 0.931646526 0.448275862 0.551724138 0.078738671  
## [36] 0.921261329 0.463448276 0.536551724 0.087613293 0.912386707

weak <- ROC\_S %>%  
 filter(decile\_cr != "0")%>%  
 select(CID,FID)%>%  
 mutate(  
 nCID =1-CID,  
 nFID = 1-FID  
 )%>%  
 select(CID,nCID,FID,nFID)%>%t()%>%  
 as.vector()  
weak

## [1] 0.066206897 0.933793103 0.007175227 0.992824773 0.150344828  
## [6] 0.849655172 0.012462236 0.987537764 0.198620690 0.801379310  
## [11] 0.019637462 0.980362538 0.252413793 0.747586207 0.028323263  
## [16] 0.971676737 0.293793103 0.706206897 0.038708459 0.961291541  
## [21] 0.337931034 0.662068966 0.045694864 0.954305136 0.371034483  
## [26] 0.628965517 0.057779456 0.942220544 0.412413793 0.587586207  
## [31] 0.068353474 0.931646526 0.448275862 0.551724138 0.078738671  
## [36] 0.921261329 0.463448276 0.536551724 0.087613293 0.912386707

ggplot(ROC, aes(x=FID, y=CID)) +   
 geom\_point(data = ROC, colour = 'black', shape = 1) +  
 geom\_point(data = ROC\_S, colour = 'blue', shape = 2) +  
 geom\_point(data = ROC\_W, colour = 'blue', shape = 3) +  
 xlim(0,.1) +  
 ylim(0,.6)+   
 ggtitle("ROC Eyewitness data (by memory strenghth)")



UVSDT\_eyewit <- function(Q, data, param.names, n.params, tmp.env){  
 n <- 8   
 mean <- Q[1]  
 sd <- Q[2]  
 cr <- c(Q[3:12]) #confidence criterion  
 v <- vector()  
 for (i in 1:length(cr)) {  
 CID <- integrate(  
 f = function(x){  
 dnorm(x,mean,sd)\*(pnorm(x,0,1)^(n-1))  
 },  
 lower = cr[i],  
 upper = Inf  
 )$value  
 FA <- (1-pnorm(cr[i])^n)/n  
 I <- c(  
 t(  
 cbind(CID, (1-CID), FA, 1-(FA))  
 )  
 )  
v <- cbind(v,I)  
 }  
 Outcomes <- c(t(v))  
 return(Outcomes)  
}  
   
  
IndObvMLE2 <- function(Q, data, param.names, n.params, tmp.env, lower.bound, upper.bound){  
 e <- UVSDT\_eyewit(Q, param.names, n.params, tmp.env)  
 LL <- -sum(data[data!=0]\*log(e[data!=0]))  
 return(LL)  
}  
  
fit\_kafc <- fit.mptinr(  
 data = weak,  
 objective = IndObvMLE2,  
 param.names = c("mu", "sigma", "cr1", "cr2", "cr3", "cr4", "cr5", "cr6", "cr7", "cr8", "cr9", "cr10"),  
 categories.per.type = c(4,4,4,4,4,4,4,4,4,4),  
 prediction = UVSDT\_eyewit,  
 lower.bound = c(0,0.1,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf),  
 upper.bound = Inf,  
 n.optim = 5,  
 starting.values = c(0,0,0,0,0,0,0,0,0,0,0,0),  
 show.messages = FALSE  
)

## Warning in fit.mptinr(data = weak, objective = IndObvMLE2, param.names = c("mu", : Optimization routine for dataset(s) 1 did not converge succesfully.  
## Error code(s): . Try use.gradient == TRUE or use output = 'full' for more information.

fit\_kafc$goodness.of.fit

## Log.Likelihood G.Squared df p.value  
## 1 -17.74057 -7.081778 18 1

fit\_kafc$parameters

## estimates lower.conf upper.conf  
## cr1 -3.9127267 NA NA  
## cr10 -4.5269027 NA NA  
## cr2 -5.4881596 NA NA  
## cr3 1.8158736 NA NA  
## cr4 -8.6043974 NA NA  
## cr5 -3.8235399 NA NA  
## cr6 -5.0133716 NA NA  
## cr7 1.7677475 NA NA  
## cr8 -8.5603719 NA NA  
## cr9 -4.0244680 NA NA  
## mu 1.3401927 NA NA  
## sigma 0.3529546 NA NA

fit\_kafc <- fit.mptinr(  
 data = strong,  
 objective = IndObvMLE2,  
 param.names = c("mu", "sigma", "cr1", "cr2", "cr3", "cr4", "cr5", "cr6", "cr7", "cr8", "cr9", "cr10"),  
 categories.per.type = c(4,4,4,4,4,4,4,4,4,4),  
 prediction = UVSDT\_eyewit,  
 lower.bound = c(0,0.1,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,-Inf),  
 upper.bound = Inf,  
 n.optim = 5,  
 starting.values = c(0,0,0,0,0,0,0,0,0,0,0,0),  
 show.messages = FALSE  
)

## Warning in fit.mptinr(data = strong, objective = IndObvMLE2, param.names = c("mu", : Optimization routine for dataset(s) 1 did not converge succesfully.  
## Error code(s): . Try use.gradient == TRUE or use output = 'full' for more information.

fit\_kafc$goodness.of.fit

## Log.Likelihood G.Squared df p.value  
## 1 -17.74057 -7.081778 18 1

fit\_kafc$parameters

## estimates lower.conf upper.conf  
## cr1 -3.9127267 NA NA  
## cr10 -4.5269027 NA NA  
## cr2 -5.4881596 NA NA  
## cr3 1.8158736 NA NA  
## cr4 -8.6043974 NA NA  
## cr5 -3.8235399 NA NA  
## cr6 -5.0133716 NA NA  
## cr7 1.7677475 NA NA  
## cr8 -8.5603719 NA NA  
## cr9 -4.0244680 NA NA  
## mu 1.3401927 NA NA  
## sigma 0.3529546 NA NA