Decision Theory: Assignment 3

Nicolas Taba (nicta839)

08/12/2021

Exercise 1

A

We calculate the weighted average payoff for each action

```
#set up constants
highlow <- c(0.4, 0.6)
wa <- c()
# setup the dataframe/matrix
data1 <- c(-50, 30, 10, -10, 80, 40, 30, -50, 20, 70, -30, -70, 100, 20, 10, -20, 0, 50, 40, 200)
df <- data.frame(matrix(data = data1, nrow = 4, ncol = 5))
print(df)
```

```
##    X1  X2  X3  X4  X5
## 1 -50  80  20 100   0
## 2  30  40  70  20  50
## 3  10  30 -30  10  40
## 4 -10 -50 -70 -20 200
```

```
# apply function to all rows
for(i in 1:4){
  max_now <- max(df[i,])
  min_now <- min(df[i,])
  wa[i] <- highlow[1] * max_now + highlow[2] * min_now
}

#output maximum output
print(wa)
```

```
## [1] 10 40 -2 38
```

```
# cat("The highest weighted average is: ", max(wa))
```

The highest payoff is for action 2. Therefore, under the weighted average criterion, this is the optimal action.

B

We calculate for each action the utility for each state of the world and multiply each utility found by the corresponding prior probability of their respective state of the world.

```
# Set up constants
probs <- c(0.1, 0.2, 0.25, 0.1, 0.35)
df1b <- data.frame(matrix(data = 0, nrow = 4, ncol = 5))
EU <- c()
# calculate in a loop
```

```
for(i in 1:4){
  for (j in 1:5){
    # log * probs
    df1b[i, j] <- log(df[i,j] + 71)
  }
  # sum up the line
  EU[i] <- sum(df1b[i, ] * probs)
}

# find the largest value
print(EU)
```

```
## [1] 4.441727 4.770221 4.378643 3.373916
```

Under the EU criterion, the optimal action is action 2.

Exercise 2

A

The states of the world are S_1 "70% red/30% blue" and S_2 "70% blue/30% red" and the actions are A_1 "guess S_1" and A_2 "guess S_2". The payoff table has the reward in the diagonal and the penalty in the anti-diagonal.

```
# setting up exercise information
lik <- matrix(c(0.7, 0.3, 0.3, 0.7), ncol = 2)
payoff <- matrix(c(5, -3, -3, 5), ncol = 2)
p_S1 <- 0.4
p_S2 <- 0.6

# Expected payoff prior
ER_S1 <- payoff[1,1] * p_S1 + payoff[2,1] * p_S2
ER_S2 <- payoff[1,2] * p_S1 + payoff[2,2] * p_S2

print(ER_S1)
```

```
## [1] 0.2
```

```
print(ER_S2)
```

```
## [1] 1.8
```

The optimal action according to our prior knowledge is S_2.

We now calculate the expected payoff in the psoterior sense

```
# setting up numerator and denominators for posterior probabilities
denom1 <- lik[1,1] * p_S1 + lik[2,1] * p_S2
denom2 <- lik[1,2] * p_S1 + lik[2,2] * p_S2

num1 <- lik[1,1] * p_S1
num2 <- lik[2,1] * p_S2
num3 <- lik[1,2] * p_S1
num4 <- lik[2,2] * p_S2

# posterior probability
post_S1_red <- num1 / denom1
post_S2_red <- num2 / denom1
post_S1_blue <- num3 / denom2
```

```
post_S2_blue <- num4 / denom2

# marginal probabilities
p_red <- lik[1,1] * p_S1 + lik[2,1] * p_S2
p_blue <- lik[1,2] * p_S1 + lik[2,2] * p_S2

# expected payoff in the posterior sense
ER_S1_red <- payoff[1,1] * post_S1_red + payoff[1,2] * post_S2_red
ER_S2_red <- payoff[2,1] * post_S1_red + payoff[2,2] * post_S2_red
ER_S1_blue <- payoff[1,1] * post_S1_blue + payoff[1,2] * post_S2_blue
ER_S2_blue <- payoff[2,1] * post_S1_blue + payoff[2,2] * post_S2_blue

ER_post <- matrix(c(ER_S1_red, ER_S1_blue, ER_S2_red, ER_S2_blue), nrow = 2)

print(ER_post)
```

```
##           [,1]      [,2]
## [1,]  1.869565 0.1304348
## [2,] -1.222222 3.2222222
```

The optimal posterior action given we pick “red” is A_1 and given that we pick “blue” is A_2.

Now we can calculate the value of sample information and the expected value of sample information.

```
# VSI
# given red
VSI_red <- ER_post[1,1] - ER_post[1,2]
# given blue
VSI_blue <- ER_post[2,2] - ER_post[2,2]

#EVSI
EVSI = VSI_red*(lik[1,1]*p_S1 + lik[2,1]*p_S2) + VSI_blue*(lik[1,2]*p_S1 + lik[2,2]*p_S2)

cat("The expected value of sample information is: ", EVSI)
```

```
## The expected value of sample information is:  0.8
```

b

We must calculate the VSI after each sample following the procedure of meeting 12 where exercise 17 was solved. We have a binomial sampling problem so we can take advantage of the density function method in R.

```
EVSI_10 <- 0
price <- 0.25
for(i in c(0:10)){
  #calculate p(y)
  p_y <- dbinom(i, 10, 0.3) * p_S2
  #calculate p(theta|y)
  post_theta_1 <- (dbinom(i, 10, lik[1,1]) * p_S1) / p_y
  post_theta_2 <- (dbinom(i, 10, lik[1,2]) * p_S2) / p_y
  #calculate ER for both options
  ER_red <- payoff[1,1] * post_theta_1 + payoff[1,2] * post_theta_2
  ER_blue <- payoff[2,1] * post_theta_1 + payoff[2,2] * post_theta_2
  #Set prior ER as ER_blue
  ER_prior <- ER_blue
  #calculate the VSI
```

```r
  VSI_temp <- max(ER_red, ER_blue) - ER_prior
  #Add to the EVSI
  EVSI_10 <- VSI_temp * p_y + EVSI_10

}
```

Now that we have the EVSI for 10 successive samples, we can calculate the expected net gain of sampling

```r
ENGS_10 <- EVSI_10 - 10 * price

cat("The ENGS is: ", ENGS_10)
```

```
## The ENGS is:  -0.008133804
```