b365 hw4

Morgan Fissel

2023-02-22

Problem 1

- a) Using this simplification, create a 3-dimensional table on age, education and vote. Show your code and your output.

```
x = read.csv("chilean_voting.csv",stringsAsFactors=FALSE, sep=",")
vote = 9        # the vote (Y,N,A,U) is 9th column
yes = (x[,vote] == "Y" & !(is.na(x[,vote])))   # boolean vector giving "Yes" votpers who are not missin
no = (x[,vote] == "N" & !(is.na(x[,vote])))
x = x[no | yes,]

# some other variables
education = 6   # education is the 6th column
gender = 4
region = 2
age = 5
x[,age] = floor(x[age]/10)
t = table(x[,c(age,education,vote)])
print(t)
```

```
## , , vote = N
##
##    education
## age   P  PS   S
##   1   8   8  52
##   2  60  99 154
##   3  47  62  90
##   4  52  33  49
##   5  53  16  27
##   6  40   5  21
##   7   6   1   4
##
## , , vote = Y
##
##    education
## age   P  PS   S
##   1   8   7  32
##   2  79  44  97
##   3  76  36  76
##   4  89  26  41
```

```
##   5  77   9  36
##   6  79   6  24
##   7  14   2   5
```

- b) Using this table create a Bayes' classifier to predict the voting status of a person given their decade and education level. You can represent your classifier as we have done in class. That is, as a table where the rows account for all possible configurations of the decade and education variables, giving the vote classification for each.

```
# Programming a simple Bayes' classifier.
# Here we assume we know the probabilities, though wouldn't in the real world
age_levels = c("1", "2", "3", "4", "5", "6", "7")
edu_levels = c("P", "S", "PS")
K = 2     # number of classes (yes or no values)
V = 21    # number of different values of our data observation could take (age_uni * edu_uni)
prior = table(x[, vote])/nrow(x)
prior = prior/sum(prior)  # the (made-up) "prior" probabilites of the different classes
Q = matrix(runif(K*V),K,V, dimnames = list(c("N", "Y"), paste(rep(edu_levels, each = length(age_levels))
for (k in 1:K) Q[k,] = Q[k,] / sum(Q[k,])
# Q is the data matrix.
# the kth row of Q (Q[k,]) is the probability distribution on the data value when
# chosen from kth class

set.seed(123)
N = 10000
correct = 0
for (i in 1:N) {
    c = sample(1:K,1,prob=prior)      # the class.  we assume this comes from prior dist
    x1 = sample(1:V,1,prob=Q[c,])  # Q[c,] gives dist of observation (x) when c is true class
    post = prior * Q[,x1]      # Q[,x] gives probabilites of x under each of different classes
    post = post / sum(post)    # normalizing so post is posterior distribution
    chat = which.max(post)     # Bayes choose class with biggest "posterior" prob (ie given data).
                        # makes sense.
    #cat(c, chat, "\n")
    if (chat == c) correct = correct+1
}
cat("prob of correct class = ", correct/N,"\n")
```

```
## prob of correct class =  0.646
```

```
# should usually beat "prior" classifier that classifies according to most likley prior prob.

t(Q)
```

```
##                N           Y
## P_1  0.033037399 0.028684886
## P_2  0.001440808 0.083574444
## P_3  0.014748753 0.042544041
## P_4  0.059684696 0.083462516
## P_5  0.092136212 0.063004375
## P_6  0.015134848 0.015595942
## P_7  0.033722992 0.016160436
```

```
## S_1  0.037014720 0.040855724
## S_2  0.054323478 0.087977217
## S_3  0.004151289 0.056161693
## S_4  0.093791785 0.033815474
## S_5  0.048631538 0.061102661
## S_6  0.009374567 0.027883883
## S_7  0.026624305 0.020844616
## PS_1 0.075627421 0.057372424
## PS_2 0.052278181 0.057170748
## PS_3 0.085739566 0.057011252
## PS_4 0.084843692 0.076046702
## PS_5 0.007414038 0.034562007
## PS_6 0.094236512 0.052204994
## PS_7 0.076043200 0.003963967
```

- c) How would the Bayes' classifier classify a female, post-secondary-educated person from the SA region in her 50's?

```
post = prior * Q[,'PS_5']
post = post / sum(post)

cat('A Female, PS educated person in her 50s, will be classified as which is higher N or Y:')
```

```
## A Female, PS educated person in her 50s, will be classified as which is higher N or Y:
```

```
post
```

```
##
##         N         Y
## 0.1801289 0.8198711
```

- d) Explain your degree of confidence in this classification and why you believe this. The degree of confidence corresponds to the value that is determined by the max value of the posterior distribution. So however confident is determined by the posterior values, and it then the corresponding classification of Yes or No. Setting the seed allows for reproducibility. Showing we can be about 90% confident in a Yes vote.

Problem 2

Continuing with Chilean voting. - a) Estimate the prior distribution on the vote (Y or N) using the data.

```
prior = table(x[, vote])/nrow(x)
prior = prior/sum(prior)
cat('Prior using the data: ', prior)
```

```
## Prior using the data:  0.5059761 0.4940239
```

- b) Separately for both the Yes and No voters, estimate the class-conditional distributions for gender, education, region, and age.

```
yes_v = x[yes,]
table_yes_gender = table(yes_v[,c('sex')])
table_yes_education = table(yes_v[,c('education')])
table_yes_region = table(yes_v[,c('region')])
table_yes_age = table(yes_v[,c('age')])
print(table_yes_gender)
```

```
##
##   F   M
## 300 316
```

```
print(table_yes_education)
```

```
##
##   P  PS   S
## 251 135 225
```

```
print(table_yes_region)
```

```
##
##   C   M   N   S  SA
## 115  15  98 176 212
```

```
print(table_yes_age)
```

```
##
##   1   2   3   4   5   6   7
##  40 181 144 101  79  62   9
```

```
no_v = x[no,]
table_no_gender = table(no_v[,c('sex')])
table_no_education = table(no_v[,c('education')])
table_no_region = table(no_v[,c('region')])
table_no_age = table(no_v[,c('age')])
print(table_no_gender)
```

```
##
##   F   M
## 262 303
```

```
print(table_no_education)
```

```
##
##   P  PS   S
## 223 105 236
```

```
print(table_no_region)
```

```
##
##   C   M   N   S  SA
## 151  18  79 134 183
```

```
print(table_no_age)
```

```
##
##   1   2   3   4   5   6   7
##  37 188 109  89  74  56  12
```

- c) How would the naive Bayes' classifier classify a female, post-secondary-educated person from the SA region in her 50s? Show the calculations clearly.

```
p_y_gender = prop.table(table_yes_gender)
p_y_education = prop.table(table_yes_education)
p_y_region = prop.table(table_yes_region)
p_y_age = prop.table(table_yes_age)
p_n_gender = prop.table(table_no_gender)
p_n_education = prop.table(table_no_education)
p_n_region = prop.table(table_no_region)
p_n_age = prop.table(table_no_age)

p = prop.table(table(x[,c('sex')]))
p_f = p[1]
p_m = p[2]
edu = prop.table(table(x[,c('education')]))
p_ps = edu['PS']
reg = prop.table(table(x[,c('region')]))
p_sa = reg['SA']
decade = prop.table(table(x[,c('age')]))
p_age = decade['5']
p_f_ps_sa_age = p_f * p_ps * p_sa * p_age
p_y = p_y_gender['F'] * p_y_education['PS'] * p_y_region['SA'] * p_y_age['5'] * prior['Y'] / p_f_ps_sa_a
p_n = p_n_gender['F'] * p_n_education['PS'] * p_n_region['SA'] * p_n_age['5'] * prior['N'] / p_f_ps_sa_a
cat('Probability of Yes classification:', p_y, '\nProbability of No classification: ',p_n)
```

```
## Probability of Yes classification: 0.5739776
## Probability of No classification:  0.4533058
```

Problem 3

- a) Simulate a boolean variable that behaves like the described medical condition.

```
probs = c(.65, .60, .57, .62, .58, .64, .67, .58, .61, .60)
n = 1000
p_cond = rbinom(n,1,0.3)
p_cond
```

```
##    [1] 1 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 1 1 1 0 0 0 1 0
##   [38] 1 0 0 0 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
```

```
##   [75] 1 1 0 0 0 1 1 0 0 0 0 0 1 0 1 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0
##  [112] 0 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0
##  [149] 0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 0 1 0 0 0 0 0 0
##  [186] 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 1 0 0
##  [223] 1 1 0 1 1 0 0 0 0 1 0 1 0 0 1 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 1 0 0 1 1 0 0
##  [260] 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 0 0
##  [297] 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0
##  [334] 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0
##  [371] 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 0 1 0 0 1 1 0 0 0 1 0 0
##  [408] 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 1 1 1 0 0 1
##  [445] 1 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0
##  [482] 1 0 1 0 0 0 0 0 0 0 1 0 1 1 0 1 1 1 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0
##  [519] 1 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1
##  [556] 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 1 1 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0
##  [593] 1 0 1 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 0 0 0 0 0 1 1 1
##  [630] 0 1 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0
##  [667] 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
##  [704] 0 1 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1
##  [741] 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 1 1 0 0 1 0 1 0 1 1 1 1 0 0 0 1 0 1 0 1 0 0 0
##  [778] 0 1 0 1 1 1 1 0 1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 1 0 0 0 0 0 0 1 0 0 1 1 1 0 0
##  [815] 1 0 1 1 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1 0 0 1 0 1 1 1 1 0 0 1 0
##  [852] 0 0 0 1 0 0 0 0 0 1 1 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 1 0 0 0 0 0 0
##  [889] 1 1 1 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0
##  [926] 1 0 0 1 0 1 0 1 1 0 0 1 1 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0
##  [963] 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0
## [1000] 0
```

- b) Generate the results of the 10 tests (which depend on whether or not the condition exists)

```
test_results = matrix(0, ncol = 10, nrow = n)
for (i in 1:n) {
  if (p_cond[i] == 1) { # Condition is present
    test_results[i, ] = rbinom(10, 1, probs)
  } else { # Condition is not present
    test_results[i, ] = rbinom(10, 1, probs)
  }
}
head(test_results,10)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    0    1    1    1    0    0    1    0    1     0
##  [2,]    0    1    0    0    1    0    1    1    0     1
##  [3,]    1    0    1    1    1    0    1    1    1     0
##  [4,]    1    0    0    0    0    0    0    1    0     0
##  [5,]    1    0    0    0    1    0    1    1    0     0
##  [6,]    1    0    0    0    1    1    1    1    1     1
##  [7,]    0    1    1    0    1    1    1    0    0     0
##  [8,]    0    0    1    1    0    1    1    0    1     1
##  [9,]    1    1    0    1    1    1    1    0    1     0
## [10,]    1    0    0    1    0    1    0    0    1     1
```

- c) Compute the posterior probability that the condition is present, given the test results.

```
prior_prob = 0.3
prob_present = probs
prob_absent = probs
present = apply(test_results, 1, function(x) prod(prob_present[x == 1]))
absent = apply(test_results, 1, function(x) prod(prob_absent[x == 0]))
tot = present * prior_prob + absent * (1 - prior_prob)

# Compute posterior probability
posterior_prob = present * prior_prob / tot

head(posterior_prob, 10)
```

```
##  [1] 0.30667675 0.27941606 0.05548675 0.89328903 0.55832741 0.06486144
##  [7] 0.29879000 0.15338742 0.07343669 0.34419635
```

- d) Classify the the instance as either "trait present" or "trait not present"

```
classification = matrix(0, ncol = 10, nrow = n)
for (i in 1:n){
  if (posterior_prob[i] >= 0.5) {
    classification[i,] = "trait present"
  } else {
    classification[i,] = "trait not present"
  }
}
head(classification,10)
```

```
##       [,1]                [,2]                [,3]
##  [1,] "trait not present" "trait not present" "trait not present"
##  [2,] "trait not present" "trait not present" "trait not present"
##  [3,] "trait not present" "trait not present" "trait not present"
##  [4,] "trait present"     "trait present"     "trait present"
##  [5,] "trait present"     "trait present"     "trait present"
##  [6,] "trait not present" "trait not present" "trait not present"
##  [7,] "trait not present" "trait not present" "trait not present"
##  [8,] "trait not present" "trait not present" "trait not present"
##  [9,] "trait not present" "trait not present" "trait not present"
## [10,] "trait not present" "trait not present" "trait not present"
##       [,4]                [,5]                [,6]
##  [1,] "trait not present" "trait not present" "trait not present"
##  [2,] "trait not present" "trait not present" "trait not present"
##  [3,] "trait not present" "trait not present" "trait not present"
##  [4,] "trait present"     "trait present"     "trait present"
##  [5,] "trait present"     "trait present"     "trait present"
##  [6,] "trait not present" "trait not present" "trait not present"
##  [7,] "trait not present" "trait not present" "trait not present"
##  [8,] "trait not present" "trait not present" "trait not present"
##  [9,] "trait not present" "trait not present" "trait not present"
## [10,] "trait not present" "trait not present" "trait not present"
##       [,7]                [,8]                [,9]
##  [1,] "trait not present" "trait not present" "trait not present"
##  [2,] "trait not present" "trait not present" "trait not present"
```

```
##  [3,] "trait not present" "trait not present" "trait not present"
##  [4,] "trait present"     "trait present"     "trait present"
##  [5,] "trait present"     "trait present"     "trait present"
##  [6,] "trait not present" "trait not present" "trait not present"
##  [7,] "trait not present" "trait not present" "trait not present"
##  [8,] "trait not present" "trait not present" "trait not present"
##  [9,] "trait not present" "trait not present" "trait not present"
## [10,] "trait not present" "trait not present" "trait not present"
##       [,10]
##  [1,] "trait not present"
##  [2,] "trait not present"
##  [3,] "trait not present"
##  [4,] "trait present"
##  [5,] "trait present"
##  [6,] "trait not present"
##  [7,] "trait not present"
##  [8,] "trait not present"
##  [9,] "trait not present"
## [10,] "trait not present"
```

- e) Keep a tally of the number of correctly identified individuals and compute the error rate of your classifier.

```
correct_predictions = 0
incorrect_predictions = 0

# Classify each instance and keep tally of correct/incorrect predictions
for (i in 1:n) {
  if (posterior_prob[i] >= 0.5) {
    classification = "trait present"
  } else {
    classification = "trait not present"
  }

  if (classification == "trait present" && p_cond[i] == 1) {
    correct_predictions = correct_predictions + 1
  } else if (classification == "trait not present" && p_cond[i] == 0) {
    correct_predictions = correct_predictions + 1
  } else {
    incorrect_predictions = incorrect_predictions + 1
  }
}

# Compute error rate
error_rate = incorrect_predictions / n

cat('Error rate of the classifier:', error_rate, '\n')
```

```
## Error rate of the classifier: 0.337
```

```
cat('Number of correct predictions:', correct_predictions, '\n')
```

```
## Number of correct predictions: 663
```

```
cat('Number of incorrect predictions:', incorrect_predictions, '\n')
```

```
## Number of incorrect predictions: 337
```

Problem 4

- a) Construct the 5-NN classifier on the iris data by creating a vector, classhat, which is the result of the classifier applied to each instance.

```
data(iris)
d = as.matrix(dist(iris[,1:4]))
n = nrow(iris)
class = as.numeric(iris[,5])
classhat = rep(0,n)
for (i in 1:n) {
    dist_i = d[i, ]
    sorted_i = sort(dist_i)
    nn_indices = which(dist_i %in% sorted_i[1:5])
    nn_classes = class[nn_indices]
    classhat[i] = names(sort(-table(nn_classes)))[1]
}
```

- b) Estimate the error rate of your classifier.

```
error = sum(class != classhat)/n
cat("Error rate estimate = ", error)
```

```
## Error rate estimate =  0.03333333
```

- c) Explain why you do or don't believe your error rate is an accurate estimate of the generalization error rate. I would say my error rate is valid estimate, however because we do not explicitly split the data into training and testing data we are more likely to to have over-fitting and without the split we can't run cross validation to say for sure. Without knowing the generalization error rate I would guess that this classifier is over-fitting a bit.

Problem 5

- Print out the figure below and construct and label the 5 regions, A, B, C, D, E, where A is the region that will be classified as a is classified; B is the region that will be classified at b is classified, etc. Done in PDF attached couldn't figure out how to get that plot into R.