Umut Akös
21202015
CS 464 -HW1

1 SLOT MACHINE

1.1) Representation**:** O : outcome, M: machine

P(M=2 | O = L) = P(M=2, O = L) / P(O=L)

= (900 / 1330) / ((60+75+828+72)/ 1330) = (900/1330) / (1035/1330)

= 0.869565217

1.2) Representation: O : outcome, M: machine

I lose on machine 1:

P(O = L | P = YOU, M = 1) = P (P = YOU, O=L, M=1) / P(P = YOU, M = 1)

= (60/1330) / (100/1330) = 0.6

I lose on machine 2:

P(O=L | P = YOU, M= 2) = P(P = YOU, O=L , M =2) / P(P = YOU, M = 2)

= (828/1330) / (1040/1330) = 0.796153846

My friend lose machine 1:

P(O = L | P = FRIEND, M = 1) = P(P = FRIEND, O = L, M=1) / P(P = FRIEND, M=1)

= (75/1330) / (100 /1330) = 0.75

My friend lose machine 2:

P(O = L | P = FRIEND, M = 2) = P(P = FRIEND, O = L, M=2) / P(P = FRIEND, M=2)

= (72/1330) / (90/ 1330) = 0.8

Result:

P(O = L | P = YOU, M = 1) < P(O = L | P = FRIEND, M = 1) 0.6 < 0.75

P(O = L | P = YOU, M = 2) < P(O = L | P = FRIEND, M = 2) 0.7.. < 0.8

My friend is more likely to lose in both machine 1 and machine 2.

1.3) Total Number of Win/ Lose = 295/1035

My Total Number of Win/ Lose = 252 / 888

Friend Total Number of Win/ Lose = 43/ 147

Probability of I win

P(O = W | P = YOU) = P(O = W | P = YOU) / P (P = YOU)

252 / (252 + 888) = 0.2211

Probability of Friend win

P(O = W | P = FRIEND) = P(O = W | P = FRIEND) / P (P = FRIEND)

43 / (43 + 147) = 0.2474

My Friend is more likely to win.

1.4) In question 1.2

1. I lose = 0.6
2. Friend win = 0.25
3. I win = 0.4
4. Friend lose = 0.75

Loss, Win, Win, Loss = 0.6 * 0.25 * 0.4 * 0.75 = 0.045

2 Rolling the dice

2.1) Representation: outcome of the blue die : b outcome of the red die : r

P(C) = P($b \neq 1, b \neq 6, r \neq 1, r \neq 2$) = P($b \neq 1, b \neq 6$) P($r \neq 1, r \neq 2$) = 4/6 *4/6 = 16 /36

P(b = 5, c = 5 | C) = P($b \neq 1, b \neq 6, b = 5, r \neq 1, r \neq 2, r = 5$) / P(C)

= 1/6*1/6 = 1/36 = (1/36) / (16/36) = 1/16

2.2) D = (b*r) %2 = 1

P(D) = 9/36

P(b = 5, r = 5 | D) = P(b = 5, r = 5, D) / P(D) still b and r is not 1,6 and 1,2

(1/9) / (9/36) = 1/36

2.3) P(b = 5, r = 5 | C) = 1/16 not equal P(b=5, r = 5)

P(b = 5, r = 5 | D) = 1/36 equals to P(b=5 , r = 5)

Second probability say that “b=5, r = 5 and D” all of them independent event so that does not effect probability but first probability, “b=5, r = 5 and C” are dependent events. The difference between this two equation is this.

3 MLE and MAP

3.1) The MLE is the value λ which maximizes

$$f(\lambda) = \prod_{i=1}^{n} e^{-\lambda} \frac{(\lambda^{xi})}{(x_i!)} = \frac{(e^{-\lambda}\lambda^{\bar{x}})^n}{(x_1!....x_n!)}$$

Where $\bar{x} = (x_1 + ... x_n)/n$, we need to find the λ which is maximizes

$$g(\lambda) = e^{\lambda}\lambda^{\bar{x}} = \exp(-\lambda + \bar{x}\ln(\lambda))$$

$$g'(\lambda) = (-1 + \bar{x}/\lambda)\, g(\lambda)$$

The maximum of g is reached when λ = $\bar{x}$. Thus the MLE of the standard deviation is $\sqrt{\bar{x}}$

3.2) λ' = argmax log $\prod_{i=1}^{n} \frac{(e^{-\lambda}\lambda^{x} i)}{(x_i!)} \lambda^{-(k+1)}$

$$\lambda = \sum_{i=1}^{n} (-\lambda + x_i(log\lambda) - \log_{(x_i)}! - (k+1)\, log\lambda + logk)$$

3.3)

4 Building a Newsgroup Classifier with Naive Bayes

4.1) We can ignore the denominator because for all j values, it has a constant value and ignoring the denominator does not change result.

4.2) 800 / 1600 = ½ = 0.5

4.3) We need 7 parameters to estimate the model.

4.4) (Test Code in the file and below)Test set accuracy is 0.5

The classifier ended up predicting with medical and MLE is bad idea because some involve models where the number of parameters increases with the number of data points (features) so this situation changes probability.

4.5) (Test Code in the file and below)

4.6)

4.7)

CODE

4.4)

```
import csv
from array import *
import math


train_features = list(csv.reader(open('question-4-train-features.csv')))
train_labels = list(csv.reader(open('question-4-train-labels.csv')))


numrows = len(train_features)    # 3 rows in your example
numcols = len(train_features[0])
#print(numrows)
#print(numcols)

spaceEstimate = [[0 for i in range(1)]  for j in range(numcols)]
medicalEstimate = [[0 for i in range(1)]  for j in range(numcols)]

#estimates
totalSpaceWordNumber = 0;
totalMedicalWordNumber = 0;


#calculate estimations
#question 4.4 for MLE
for j in range(0,numcols-1):
    for i in range(0,numrows-1):
        if train_labels[i][0] == '0': #for each madical newsmail
            medicalEstimate[j][0] = medicalEstimate[j][0] + int(train_features[i][j])
            totalMedicalWordNumber = totalMedicalWordNumber + int(train_features[i][j])
```

```
            elif train_labels[i][0] == '1': # for each space document
                spaceEstimate[j][0] = spaceEstimate[j][0] + int(train_features[i][j])
                totalSpaceWordNumber = totalSpaceWordNumber + int(train_features[i][j])

#calculation of MLE
medicalEstimate = map(lambda x: x/totalMedicalWordNumber, medicalEstimate)
spaceEstimate = map(lambda x: x/totalSpaceWordNumber, spaceEstimate)



test_features = list(csv.reader(open('question-4-test-features.csv')))
test_labels = list(csv.reader(open('question-4-test-labels.csv')))

numrows1 = len(test_labels)    # 3 rows in your example
numcols1 = len(test_labels[0])

medical_number = 0;
space_number = 0;
medical_probability = 0.0
space_probability = 0.0

for i in range (0,numrows1-1):
    if test_labels[i][0] == '0':
        medical_number = medical_number + 1
    elif test_labels[i][0] == '1':
        space_number = space_number + 1

x = [[0 for i in range(1)]  for j in range(numrows1)]
y = [[0 for i in range(2)]  for j in range(2)]

for i in range (0,numrows1-1):
    space_probability = math.log10(space_number/(space_number + medical_number))
```

```
        medical_probability = math.log10(medical_number/(space_number + medical_number))
        for j in range(0,numcols1-1):
            if test_features[i][j] != '0':
                medical_probability = medical_probability + test_features[i][j] *
math.log10(medicalEstimate[j][0])
                space_probability = space_probability + test_features[i][j] *
math.log10(spaceEstimate[j][0])
        if medical_probability >= space_probability:
            x[i][0] = 0
        elif medical_probability < space_probability:
            x[i][0] = 1



#Calculate accuracy
for i in range (0,numrows1-1):
        if x[i][0] == 1 and test_labels[i][0] == '1':
            y[0][0] = y[0][0] +1;
        elif x[i][0] == 0 and test_labels[i][0] == '0':
            y[1][1] = y[1][1] +1;
        elif x[i][0] == 1 and test_labels[i][0] == '0':
            y[0][1] = y[0][1] +1;
        elif x[i][0] == 0 and test_labels[i][0] == '1':
            y[1][0] = y[1][0] +1;



accurate = y[0][0] + y[1][1]
accuracy = accurate / 400;

print(accuracy)
```

4.5)

```
import csv
from array import *
import math
import numpy as np

#########TRAIN#############
train_features = list(csv.reader(open('question-4-train-features.csv')))
train_labels = list(csv.reader(open('question-4-train-labels.csv')))



numrows = len(train_features)
numcols = len(train_features[0])
#print(numrows)
#print(numcols)

spaceEstimate = [[0 for i in range(1)]  for j in range(numcols)]
medicalEstimate = [[0 for i in range(1)]  for j in range(numcols)]

#estimates
totalSpaceWordNumber = 0;
totalMedicalWordNumber = 0;



#calculate estimations
#question 4.4 for MLE
for j in range(0,numcols-1):
        for i in range(0,numrows-1):
                if train_labels[i][0] == '0': #for each madical newsmail
                        medicalEstimate[j][0] = medicalEstimate[j][0] + int(train_features[i][j])
                        totalMedicalWordNumber = totalMedicalWordNumber + int(train_features[i][j]) + 1
```

```
            elif train_labels[i][0] == '1': # for each space document
                spaceEstimate[j][0] = spaceEstimate[j][0] + int(train_features[i][j])
                totalSpaceWordNumber =  (totalSpaceWordNumber + int(train_features[i][j])) + 1

#calculation of MLE
medicalEstimate  = np.divide(medicalEstimate,  totalMedicalWordNumber+(numcols*numrows))
spaceEstimate  = np.divide(spaceEstimate,  totalSpaceWordNumber + (numrows*numcols))
########TEST##############
test_features = list(csv.reader(open('question-4-test-features.csv')))
test_labels = list(csv.reader(open('question-4-test-labels.csv')))

numrows1 = len(test_labels)
numcols1 = len(test_labels[0])

medical_number = 0;
space_number = 0;
medical_probability = 0.0
space_probability = 0.0

for i in range (0,numrows1-1):
    if test_labels[i][0] == '0':
        medical_number = medical_number + 1
    elif test_labels[i][0] == '1':
        space_number = space_number + 1

x = [[0 for i in range(1)]  for j in range(numrows1)]
y = [[0 for i in range(2)]  for j in range(2)]

for i in range (0,numrows1-1):
    space_probability = math.log10(space_number/(space_number + medical_number))
    medical_probability = math.log10(medical_number/(space_number + medical_number))
```

```
        for j in range(0,numcols1):
            if test_features[i][j] != '0':
                if medicalEstimate[j][0] != 0:
                    medical_probability = medical_probability + int(test_features[i][j]) *
math.log(medicalEstimate[j][0])
                    #print(medical_probability)
                    space_probability = space_probability + int(test_features[i][j]) *
math.log(spaceEstimate[j][0])
                    #print(space_probability)
        if medical_probability >= space_probability:
            x[i][0] = 0
        elif medical_probability < space_probability:
            x[i][0] = 1



#Calculate accuracy
for i in range (0,numrows1-1):
        if x[i][0] == 1 and test_labels[i][0] == '1':
            y[0][0] = y[0][0] +1;
        elif x[i][0] == 0 and test_labels[i][0] == '0':
            y[1][1] = y[1][1] +1;
        elif x[i][0] == 1 and test_labels[i][0] == '0':
            y[0][1] = y[0][1] +1;
        elif x[i][0] == 0 and test_labels[i][0] == '1':
            y[1][0] = y[1][0] +1;


accurate = y[0][0] + y[1][1]
accuracy = accurate / 400;
```

```
print(accuracy)
```