SENG 474 DATA MINING ASSIGMENT #1 V00804639 Honghu Lin

QUESTION # 1.1

Honghu, Lin V00804639

$E(f(x)) = \sum_i P(x_i) * f(x_i) = -(\frac{p}{q})\log_2(\frac{p}{q}) - (\frac{1-p}{q})\log_2(\frac{1-p}{q})$

$I(x_i) = -\log_2 P(x_i)$ $\overset{AE}{\Rightarrow}$ Sum of Respective E multiply by #/#

$H(x) = E(I(X)) = \sum_i P(x_i) I(x_i) = -\sum_i P(x_i)\log_2 P(x_i)$

choose Lowest entropy of the children nodes

Q1.1 There are 24 data set

- Age ⇒ 8 young, 8 pre-presbyopic, 8 presbyopic
- spectacle-prescrip ⇒ 12 myope, 12 hypermetrope
- astigmatism ⇒ 12 no, 12 yes
- tear-prod-rate ⇒ 12 reduced, 12 normal
- Contact-lenses ⇒ 5 soft, 4 hard, 15 none

Age entropy Calculation

8 Age = young $E(2/8, 2/8, 4/8) = -\frac{2}{8}\log_2(\frac{2}{8}) - 2/8\log_2(2/8) - 4/8\log_2(4/8)$ 1.5

8 Age = pre-presbyopic $E(2/8, 1/8, 5/8) = 1.2988$..

8 Age = presbyopic $E(1/8, 1/8, 6/8) = 1.06128$

AE AGE $= 1.5 \times 8/24 + 1.2988 \times 8/24 + 1.0613 \times 8/24 =$ 1.2867

12 Spectacle = myope $E(2/12, 3/12, 7/12) = 1.3844$

12 Spectacle = hypermetrope $E(3/12, 1/12, 8/12) = 1.1887$

AE spectacle $= 1.3844 \times 1/2 + 1.1887 \times 1/2 =$ 1.28655

12 astigmatism = no $E(5/12, 0/12, 7/12) = 0.9799$

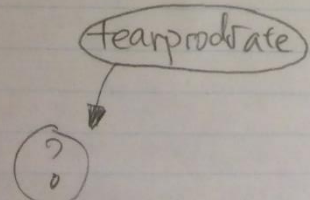
12 astigmatism = yes $E(0/12, 4/12, 8/12) = 0.9183$

AE astigmatism $= 0.9799 \times 1/2 + 0.9183 \times 1/2 =$ 0.9491

12 tearprodrate = reduced $E(0/12, 0/12, 12/12) = 0$

12 tearprodrate = normal $E(5/12, 4/12, 3/12) = 1.5546$

AE tearprodrate $= 1.5546 \times 1/2 =$ 0.7773 Smallest Value



SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

"12"

Continuing to split (for TearProdrate = Reduced)

Age = young E(0/4, 0/4, 4/4) = 0

Age = pre-presbyopic E(0/4, 0/4, 4/4) = 0

Age = presbyopic E(0/4, 0/4, 4/4) = 0

AE AGE = 0+0+0 = 0

Spectacle = myope E(0/6, 0/6, 6/6) = 0

spectacle = hypermetrope E(0/6, 0/6, 6/6) = 0

AE Spectacle = 0+0 = 0

astigmatism = no E(0/6, 0/6, 6/6) = 0

astigmatism = yes E(0/6, 0/6, 6/6) = 0

AE astigmatism = 0+0 = 0

"12"

Continuing to split (for TearProdRate = NORMAL)

Age = young E(2/4, 2/4, 0/4) = 1

Age = pre-presbyopic E(2/4, 1/4, 1/4) = 1.5

Age = presbyopic E(1/4, 1/4, 2/4) = 1.5

AE Age = 1(4/12) + 1.5(4/12) + 1.5(4/12) = 1.3333

spectacle = myope E(2/6, 3/6, 1/6) = 1.4591

spectacle = hypermetrope E(3/6, 1/6, 2/6) = 1.4591

AE spectacle = 1.4591(6/12) + 1.4591(6/12) = 1.4591

astigmatism = no E(5/6, 0/6, 1/6) = 0.6500

astigmatism = yes E(0/6, 4/6, 2/6) = 0.9183

AE astigmatism = 0.6500(6/12) + 0.9183(6/12) = 0.7842

Smallest AE

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

Continuing to Split (for spectacle = myope)

Age = young $E(1/1, 0/1, 0/1) = 0$

Age = pre-presbyopic $E(1/1, 0, 0) = 0$

Age = presbyopic $E(0, 0, 1) = 0$

AE Age $= 0+0+0 = 0$

Continuing to split (for spectacle = hypermetrope)

Age = young $E(1, 0, 0) = 0$

Age = pre-presbyopic $E(1, 0, 0) = 0$

Age = presbyopic $E(1, 0, 0) = 0$

AE Age $= 0$

- <u>Same AE</u> for TearProbrate → Astigmatism → Spectacle.

Continuing to split (for spectacle = myope)

AGE = young $E(0, 1, 0) = 0$

AGE = pre-presbyopic $E(0, 1, 0) = 0$

AGE = presbyopic $E(0, 1, 0) = 0$

AE AGE $= 0$

Continuing to split (for spectacle = hypermetrope)

AGE = young $E(0, 1, 0) = 0$

AGE = pre-presbyopic $E(0, 0, 1) = 0$

AGE = presbyopic $E(0, 0, 1) = 0$

AE AGE $= 0$

SEENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

Continuing to split (for astigmatism = no)

Age = young $E(2/2, 0/2, 0/2) = 0$

Age = pre-presbyopic $E(2/2, 0/2, 0/2) = 0$

Age = presbyopic $E(1/2, 0/2, 1/2) = 1$

AE AGE = $1(2/6) = 1/3 = 0.3333$

spectacle = myope $E(2/2, 0/2, 0/2) = 0$

Spectacle = hypermetrope $E(3/3, 0/2, 0/2) = 0$

AE spectacle = $0 + 0 = 0$ Small Value

Continuing to split (for astigmatism = yes)

Age = Young $E(0/2, 2/2, 0/2) = 0$

Age = pre-presbyopic $E(0/2, 1/2, 1/2) = 1$

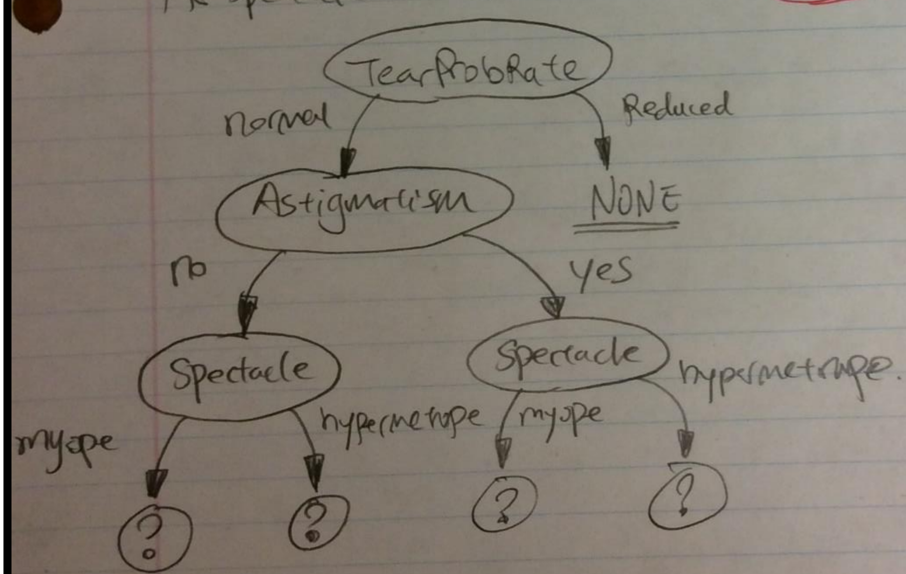
Age = presbyopic $E(0/2, 1/2, 1/2) = 1$

AE AGE = $2/6 + 2/6 = 4/6 = 2/3 = 0.6667$

spectacle = myope $E(0/3, 3/3, 0/3) = 0$

Spectacle = hypermetrope $E(0/3, 1/3, 2/3) = 0.9183$

AE spectacle = $0 + 0.9183(\frac{1}{2}) = 0.45915$ Small Value.



SEENG 474 DATA MINING ASSIGMENT #1

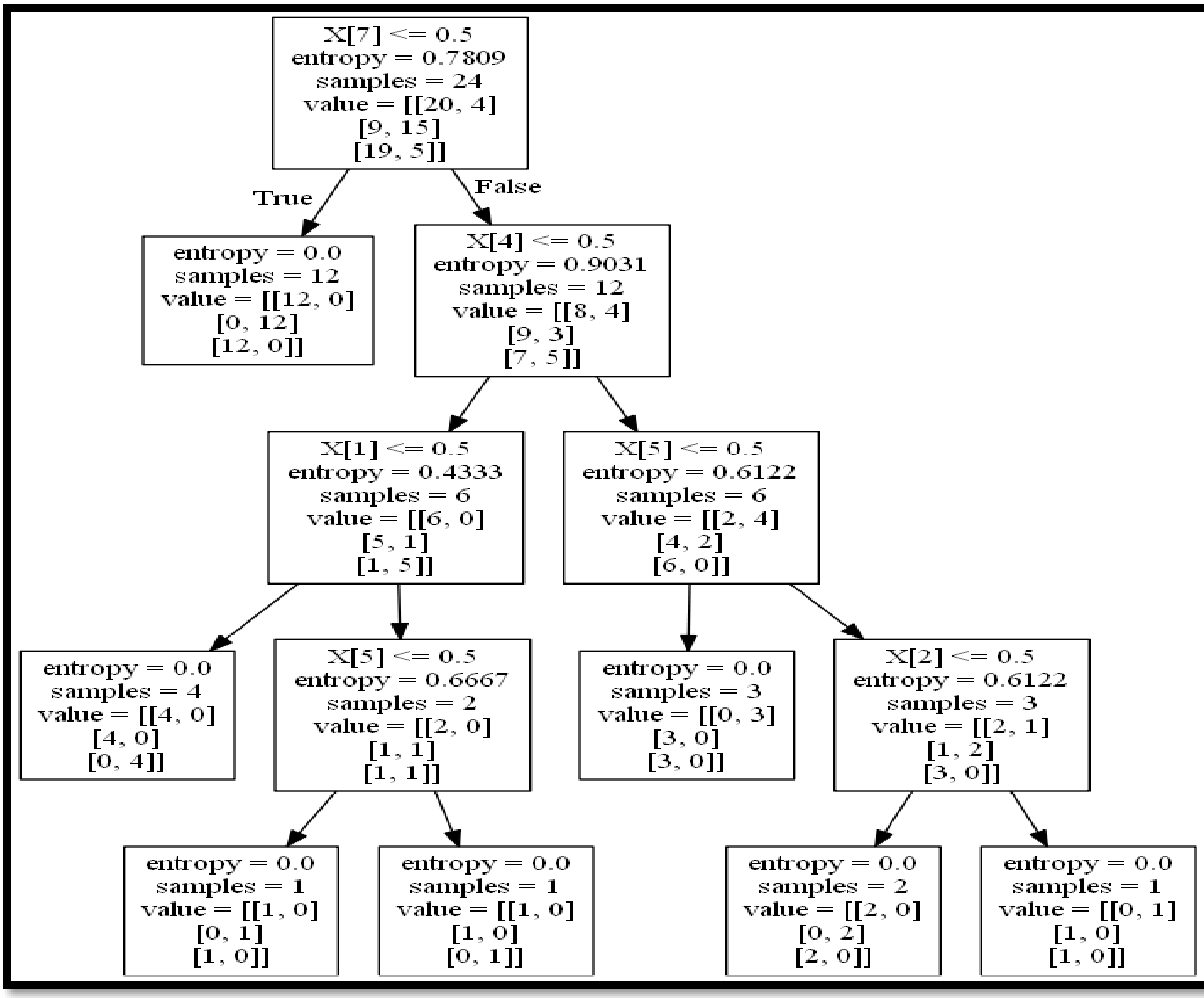
V00804639 Honghu Lin

QUESTION # 1.2

```
from sklearn import tree
from sklearn.externals.six import StringIO
from util2 import Arff2Skl
from IPython.display import Image
import graphviz
import pydot
import pydotplus

cvt = Arff2Skl('contact-lenses.arff')
label = cvt.meta.names()[-1]

X, Y = cvt.transform(label)

#print(X)# young ...  normal
#print(Y)# hard none soft

clf = tree.DecisionTreeClassifier(criterion = 'entropy')
clf = clf.fit(X,Y)
tree.export_graphviz(clf,out_file='tree1.dot')
```

SEG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin



Conclusion: For my tree my entropy is different from what I saw from the picture generated by scikit-learn, for instance you can easily tell from before the first split. My entropy is like 0.7773 for AE temperature, whereas in scikit learn I observed 0.7809 instead. I think the reason is because the calculation method is actually used differently from what I learned in class. From scikit-learn it used CART. CART will make sure the tree stay in balance.

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

QUESTION # 2

- NOTE: I did both smoothing and non-smoothing,
- NOTE: SMOOTHING IS THE **SECOND** ONE

AGE						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
YOUNG	2	2	4	2/5	2/4	4/15
PRE-PRESBYOPIC	2	1	5	2/5	1/4	5/15
PRESBYOPIC	1	1	6	1/5	1/4	6/15
TOTAL	5	4	15	100%	100%	100%

SPECTACLE-PRESCRIP						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
MYOPE	2	3	7	2/5	3/4	7/15
HYPERMETROPE	3	1	8	3/5	1/4	8/15
TOTAL	5	4	15	100%	100%	100%

ASTIGMATISM						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
NO	5	0	7	5/5	0/4	7/15
YES	0	4	8	0/5	4/4	8/15
TOTAL	5	4	15	100%	100%	100%

TEAR-PROD-RATE						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
REDUCED	0	0	12	0/5	0/4	12/15
NORMAL	5	4	3	5/5	4/4	3/15
TOTAL	5	4	15	100%	100%	100%

CLASSIFY		P(SOFT)/P(HARD)/P(NONE)
SOFT	5	5/24
HARD	4	4/24
NONE	15	15/24
TOTAL	24	100%

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

NOTE: This is the one WITHOUT smoothing

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |SOFT) = 2/5
> P (SPECTACLE-PRESCRIP = hypermetrope |SOFT) = 3/5
> P (ASTIGMATISM = yes |SOFT) = 0/5
> P (TEAR-PROD-RATE = reduced |SOFT) = 0/5
> P (CONTACT-LENSES = 5) = 5/24

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |HARD) = 1/4
> P (SPECTACLE-PRESCRIP = hypermetrope |HARD) = 1/4
> P (ASTIGMATISM = yes |HARD) = 4/4
> P (TEAR-PROD-RATE = reduced |HARD) = 0/4
> P (CONTACT-LENSES = 4) = 4/24

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |NONE) = 5/15
> P (SPECTACLE-PRESCRIP = hypermetrope |NONE) = 8/15
> P (ASTIGMATISM = yes |NONE) = 8/15
> P (TEAR-PROD-RATE = reduced |NONE) = 12/15
> P (CONTACT-LENSES = 15) = 15/24

P (X| CONTACT-LENSES = SOFT) P (CONTACT-LENSES = SOFT) = 0
P (X| CONTACT-LENSES = HARD) P (CONTACT-LENSES = HARD) = 0
P (X| CONTACT-LENSES = NONE) P (CONTACT-LENSES = NONE)
= 5*8*8*12/15/15/15/15*15/24 = 0.0474074074

P(X) = P (AGE = pre-presbyopic)*P (SPECTACLE-PRESCRIP = hypermetrope)*P (ASTIGMATISM = yes)*P (TEAR-PROD-RATE = reduced) = 8*12*12*12/24/24/24/24 = 0.04166666667

Divide the results by P(X) = 0.0474074074 / 0.04166666667 = **1.13777777751**

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

Laplace-k estimate of P (B|A) condition on A

- Random Variable B
- A is fixed

AGE – LA placed (SMOOTHING ONE !!)						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
YOUNG	2 + 1 = 3	2 + 1 = 3	4 + 1 = 5	3/8	3/7	5/18
PRE-PRESBYOPIC	2 + 1 = 3	1 + 1 = 2	5 + 1 = 6	3/8	2/7	6/18
PRESBYOPIC	1 + 1 = 2	1 + 1 = 2	6 + 1 = 7	2/8	2/7	7/18
TOTAL	5 + 3 = 8	4 + 3 = 7	15 + 3 = 18	100%	100%	100%

SPECTACLE-PRESCRIP – LA placed(SMOOTHING						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
MYOPE	2 + 1 = 3	3 + 1 = 4	7 + 1 = 8	3/7	4/6	8/17
HYPERMETROPE	3 + 1 = 4	1 + 1 = 2	8 + 1 = 9	4/7	2/6	9/17
TOTAL	5 + 2 = 7	4 + 2 = 6	15 + 2 = 17	100%	100%	100%

ASTIGMATISM – LA placed(SMOOTHING						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
NO	5 + 1 = 6	0 + 1 = 1	7 + 1 = 8	6/7	1/6	8/17
YES	0 + 1 = 1	4 + 1 = 5	8 + 1 = 9	1/7	5/6	9/17
TOTAL	5 + 2 = 7	4 + 2 = 6	15 + 2 = 17	100%	100%	100%

TEAR-PROD-RATE – LA placed(SMOOTHING						
	SOFT	HARD	NONE	P(SOFT)	P(HARD)	P(NONE)
REDUCED	0 + 1 = 1 P (x | c)	0 + 1 = 1	12 + 1 = 13	1/7	1/6	13/17 P (x)
NORMAL	5 + 1 = 6	4 + 1 = 5	3 + 1 = 4	6/7	5/6	4/17
TOTAL	5 + 2 = 7	4 + 2 = 6	15 + 2 = 17	100%	100%	100%

CLASSIFY		P(SOFT)/P(HARD)/P(NONE)
SOFT	6	6/27 P(c)
HARD	5	5/27
NONE	16	16/27
TOTAL	27	100%

P (c|x) = P (x|c) * P (c) / P (x)

Posterior prob — likelihood — class Prior Prob — Predictor Prior Prob

P (c|X) = P (x1|c) * P (x2|c) * P (x3|c) *….* P (xn|c) *P(c)

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

NOTE: This is the one WITH smoothing of value 1

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |SOFT) = 3/8
> P (SPECTACLE-PRESCRIP = hypermetrope |SOFT) = 4/7
> P (ASTIGMATISM = yes |SOFT) = 1/7
> P (TEAR-PROD-RATE = reduced |SOFT) = 1/7
> P (CONTACT-LENSES = 7) = 6/27

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |HARD) = 2/7
> P (SPECTACLE-PRESCRIP = hypermetrope |HARD) = 2/6
> P (ASTIGMATISM = yes |HARD) = 5/6
> P (TEAR-PROD-RATE = reduced |HARD) = 1/6
> P (CONTACT-LENSES = 6) = 5/27

Probability that is pre-presbyopic, hypermetrope, yes, and reduced:
> P (AGE = pre-presbyopic |NONE) = 6/18
> P (SPECTACLE-PRESCRIP = hypermetrope |NONE) = 9/17
> P (ASTIGMATISM = yes |NONE) = 9/17
> P (TEAR-PROD-RATE = reduced |NONE) = 13/17
> P (CONTACT-LENSES = 17) = 16/27

P (X| CONTACT-LENSES = SOFT) P (CONTACT-LENSES = SOFT)
= 3*4*1*1*6/8/7/7/7/27 = 0.00097181729

P (X| CONTACT-LENSES = HARD) P (CONTACT-LENSES = HARD)
= 2*2*5*1*5/7/6/6/6/27 = 0.00244953948

P (X| CONTACT-LENSES = NONE) P (CONTACT-LENSES = NONE)
= 6*9*9*13*16/18/17/17/17/27 = 0.04233665784

ALPHA = 1/P(x) => 1 / (0.00097181729 + 0.00244953948 + 0.04233665784)
P(x) = 21.8540950372

Divide the results by P(X) = 0.00097181729 * 21.8540950372 = 0.02123818741 = 2.1%
Divide the results by P(X) = 0.00244953948 * 21.8540950372 = 0.05353246859 = 5.3%
Divide the results by P(X) = 0.04233665784 * 21.8540950372 = 0.92522934399 = 92.5%

Conclusion: since none has the most percentage, so none is most likely to be classified as pre-presbyopic, hypermetrope, yes, and reduced.

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

QUESTION # 3.1

```
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.datasets import fetch_20newsgroups
import logging
import sys
from time import time
from math import log

class MyBayesClassifier():
    def __init__(self, smooth=1):
        self._smooth = smooth # This is for add one smoothing, don't forget!
        self._feat_prob = []
        self._class_prob = []
        self._Ncls = []
        self._Nfeat = []

    def train(self, X, y):

        # Add x number of label lists
        for addSpace in range(max(y)+1):
            self._Nfeat.append([])

        # Put according array to defined label lists
        for i, yVal in enumerate(y):
            self._Nfeat[yVal].append(X[i])

        # For each label lists record num to Ncls
        for eachLabel in self._Nfeat:
            self._Ncls.append(len(eachLabel))

        # Calculate smoothing prob to class_prob
        for eachNum in self._Ncls:
            prob = float(eachNum + self._smooth) / float(len(X) + 2 * self._smooth)
            self._class_prob.append(prob)
```

```python
            prob = float(eachNum + self._smooth) / float(len(X) + 2 * self._smooth)
            self._class_prob.append(prob)

        # Calculate smoothing prob to feat_prob
        for eachLabel in self._Nfeat:
            L = []
            # iterate by index
            for eachIndex in range(len(eachLabel[0])):
                counter = 0

                # Count feature by column
                for eachRow in range(len(eachLabel)):
                    if eachLabel[eachRow][eachIndex] > 0:#prob for 0
                        counter = counter + 1

            # Calculate feat_prob array for each Label
                L.append(float(counter + self._smooth) /float(len(eachLabel) + (self._smooth * 2)))
            self._feat_prob.append(L)

        #print(self._feat_prob[1])

        #print(self._feat_prob)  # prob of feature for each label

        return

    def predict(self, X):
        L = [] * len(X)

        for row in X:
            prob = -float('inf')
            finalLabel = None
            for eachLabel in range(len(self._feat_prob)):
                sum = log(self._class_prob[eachLabel])
                for eachIndex in range(len(row)):
                    if row[eachIndex] > 0:
                        sum += np.log(self._feat_prob[eachLabel][eachIndex])
                    else:
                        sum += np.log(1 - self._feat_prob[eachLabel][eachIndex])
```

```
                        sum += np.log(1 - self._feat_prob[eachLabel][eachIndex])
                if prob < sum:
                    prob = sum
                    finalLabel = eachLabel
            L.append(finalLabel)

        return L

categories = [
        'alt.atheism',
        'talk.religion.misc',
        'comp.graphics',
        'sci.space',
    ]
remove = ('headers', 'footers', 'quotes')

data_train = fetch_20newsgroups(subset='train', categories=categories,
                                shuffle=True, random_state=42,
                                remove=remove)

data_test = fetch_20newsgroups(subset='test', categories=categories,
                               shuffle=True, random_state=42,
                               remove=remove)
print('data loaded')

y_train, y_test = data_train.target, data_test.target
#print(y_train)[1 3 2...,1 0 1] (2034,0)
#print(y_test) [2 1 1...,3 1 1] (1353,0)

print("Extracting features from the training data using a count vectorizer")
t0 = time()
vectorizer = CountVectorizer(stop_words='english', binary=False)#, analyzer='char', ngram_range=(1,3))


X_train = vectorizer.fit_transform(data_train.data).toarray() #(2034, 26576)
X_test = vectorizer.transform(data_test.data).toarray()#(1353, 26576)
feature_names = vectorizer.get_feature_names() # len() is 26576 features
```

```
for x in range(100):

    alpha = 0.01 + (x*0.01)
    clf = MyBayesClassifier(alpha)
    clf.train(X_train,y_train)

    y_pred = clf.predict(X_test)
    #print(y_pred)

    print ('%f %f \n' %(alpha, np.mean((y_test-y_pred)==0)))
```

SEENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin

QUESTION # 3.2

jupyter Untitled Last Checkpoint: 7 minutes ago (autosaved)

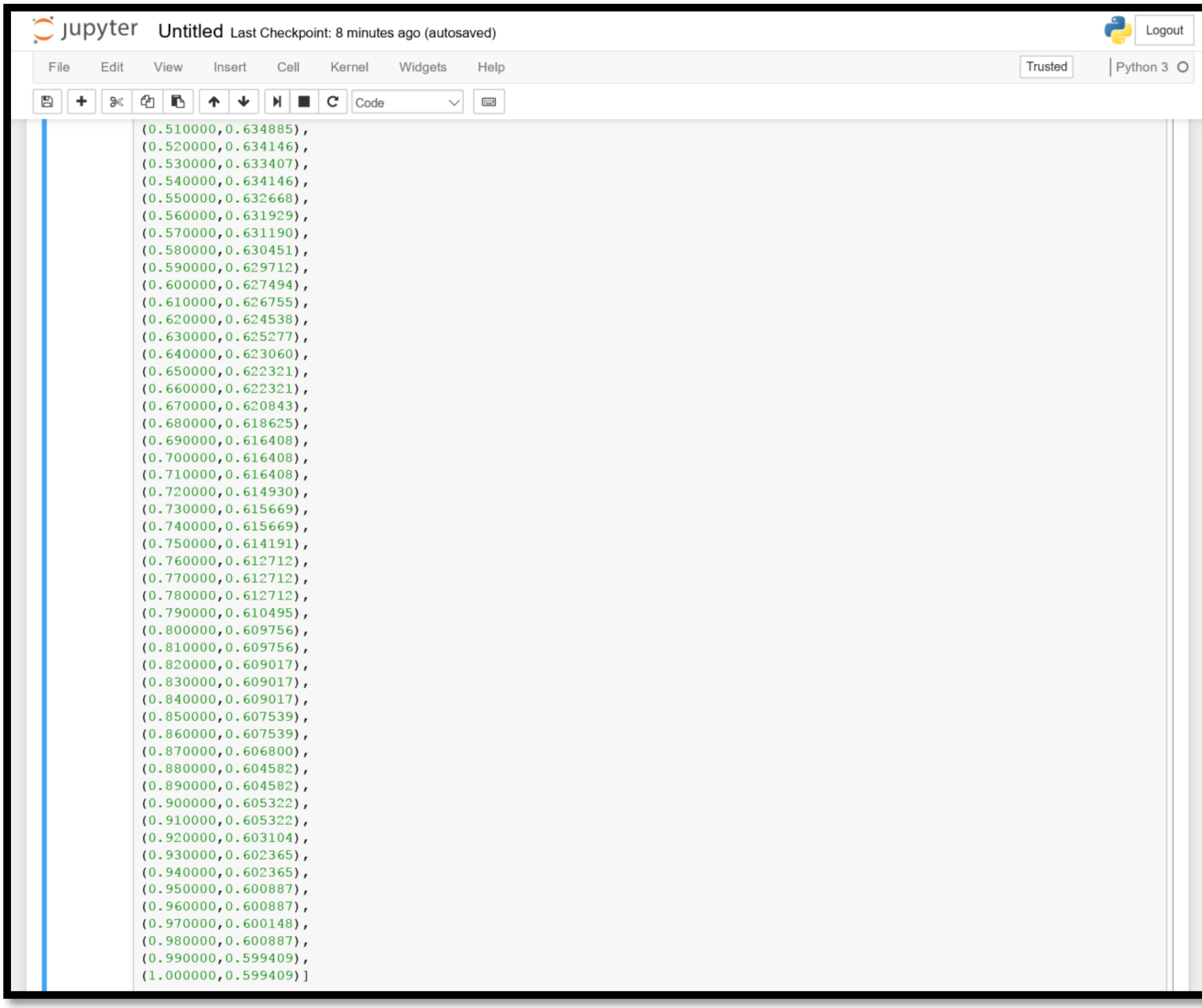
File Edit View Insert Cell Kernel Widgets Help

In [17]:

```
import matplotlib.pyplot as plt
import matplotlib.pyplot
List =[(0.010000,0.722099),
(0.020000,0.715447),
(0.030000,0.708056),
(0.040000,0.704361),
(0.050000,0.701404),
(0.060000,0.702882),
(0.070000,0.702143),
(0.080000,0.699926),
(0.090000,0.697709),
(0.100000,0.696970),
(0.110000,0.696231),
(0.120000,0.696231),
(0.130000,0.692535),
(0.140000,0.693274),
(0.150000,0.691057),
(0.160000,0.689579),
(0.170000,0.688101),
(0.180000,0.685144),
(0.190000,0.683666),
(0.200000,0.680710),
(0.210000,0.679231),
(0.220000,0.678492),
(0.230000,0.677014),
(0.240000,0.672579),
(0.250000,0.671101),
(0.260000,0.670362),
(0.270000,0.670362),
(0.280000,0.668884),
(0.290000,0.666667),
(0.300000,0.665928),
(0.310000,0.665188),
(0.320000,0.665188),
(0.330000,0.664449),
(0.340000,0.664449),
(0.350000,0.662232),
(0.360000,0.657797),
(0.370000,0.655580),
(0.380000,0.653363),
(0.390000,0.652624),
(0.400000,0.650407),
(0.410000,0.650407),
(0.420000,0.650407),
(0.430000,0.648928),
(0.440000,0.647450),
(0.450000,0.643755),
(0.460000,0.642276),
(0.470000,0.641537),
(0.480000,0.641537),
(0.490000,0.636364),
(0.500000,0.635625),
```

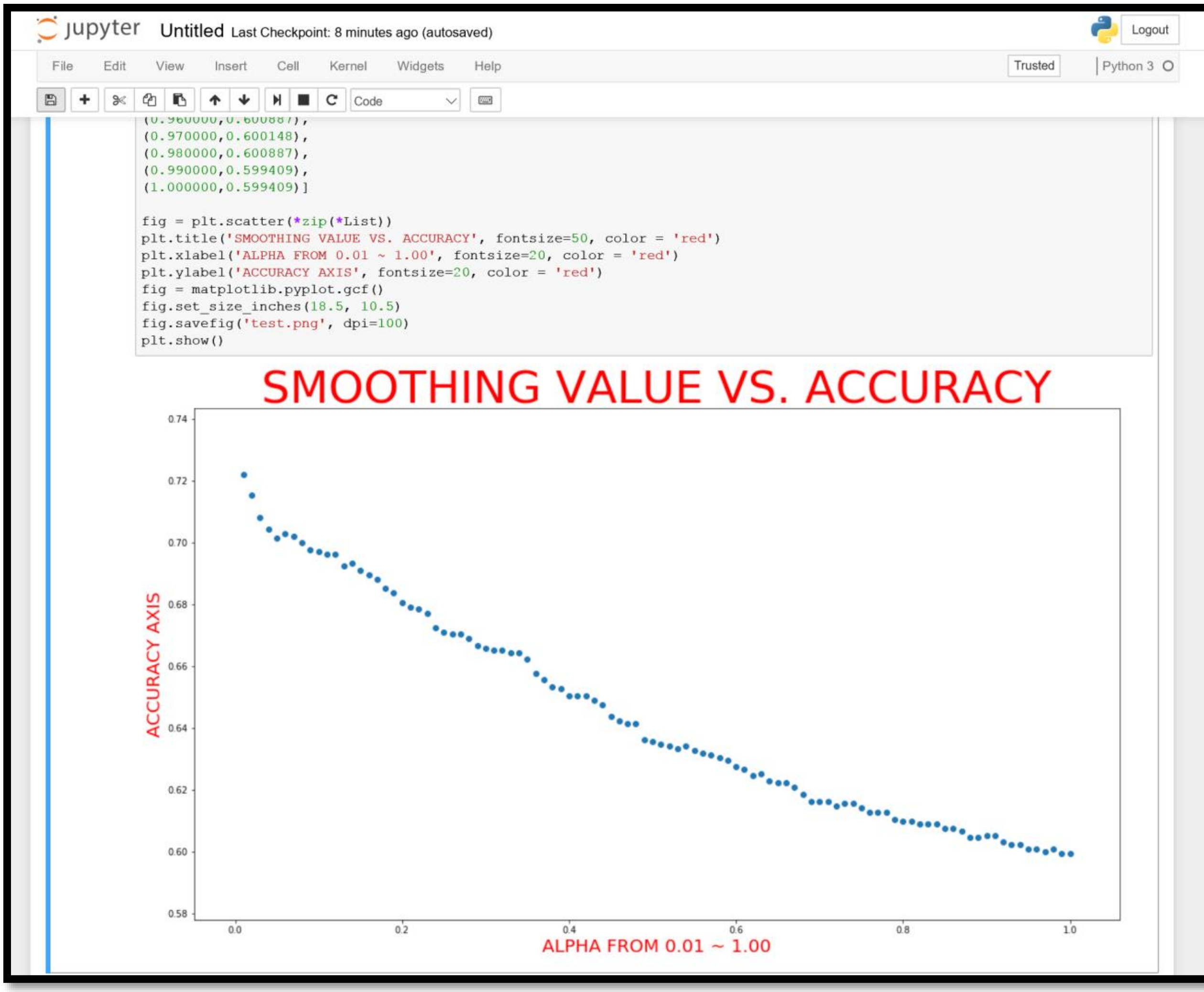
SEENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin



SENG 474 DATA MINING ASSIGMENT #1

V00804639 Honghu Lin



Conclusion: It can be tell from the graph that the plot reaches a maximum value of 0.715447 when the alpha is 0.01. As we increment the value by 0.01, even though accuracy will fluctuate at some values. Yet the main trend is going to decrease finally when we reach a minimum of 0.559409 when alpha is 1.00.