Assignment 5: Naive Bayes Classifier for Text Classification

UVA CS 6316 : Machine Learning (Fall-2019)

Vanamala Venkataswamy (vv3xu), 11-27-2019

1. Naive Bayes Classifier for Movie Review Text Classification

(Q1) You are required to implement the first choice of preprocessing described as following.

For this step, I implemented both Choice-1 and Choice-2.

Choice-1 steps:

- Read documents into text
- Split the text into words based on spaces
- Match the words with the dictionary words replacing similar words (love = loved, loving). All the words not in the dictionary are counted an Unknown (UKN)
- Use relative frequency of **wi** in mega-document
- Create a mega-document by concatenating all documents **d_1** to **d_T**

Choice-2 (NLTK) steps (**Extra Credit)**:

- Read documents into text
- Split the words using RegexpTokenizer(r'\w+') and tokenize the words using tokenizer.tokenize(text)
- Get English stopwords, using stopwords.words("english")
- Clean the text by removing stop words
- Stemming the text using PorterStemmer()
- Use relative frequency of **wi** in mega-document
- Create a mega-document by concatenating all documents **d_1** to **d_T**
- Create a vocabulary of 2000 words from movie_reviews package in NLTK

(Q2) You are required to provide the following function to convert a text document into a feature vector:

```
BOWDj = transfer(fileDj, vocabulary, choice)
```

Please refer Code

Modified the function definition to take choice as function parameter.

(Q3) Read in the training and test documents into BOW vector representations using

```
Xtrain, Xtest, ytrain, ytest = loadData(<Absolute path to data>)
```

Please refer Code.

(Q4) You are required to provide the following function (and module) for grading:
`thetaPos,thetaNeg = naiveBayesMulFeaturetrain(Xtrain,ytrain)`

Please refer Code

(Q5) Provide the resulting value of thetaPos and thetaNeg into the writeup.

Choice-1 Thetas:

```
thetaPos = [9.93883792e-04 1.55453619e-03 1.36595311e-02 7.64525994e-04
 6.88073394e-04 2.52293578e-03 2.34454638e-03 3.23649337e-03
 2.16615698e-03 2.93068298e-03 1.78389399e-03 2.39551478e-03
 1.68195719e-03 1.80937819e-03 1.86034659e-03 2.67584098e-03
 1.88583078e-03 2.98165138e-03 2.08970438e-03 2.67584098e-03
 5.70846075e-03 2.01325178e-03 2.08970438e-03 2.44648318e-03
 2.44648318e-03 2.70132518e-03 2.54841998e-05 4.38328236e-03
 2.47196738e-03 4.40876656e-03 2.67584098e-03 2.08970438e-03
 3.16004077e-03 2.87971458e-03 2.47196738e-03 1.27420999e-03
 2.29357798e-03 4.81651376e-03 8.91946993e-04 3.08358818e-03
 3.56778797e-03 2.21712538e-03 3.61875637e-03 3.46585117e-03
 3.16004077e-03 3.21100917e-03 3.69520897e-03 3.26197757e-03
 2.77777778e-03 4.38328236e-03 3.49133537e-03 3.97553517e-03
 6.93170234e-03 2.11518858e-03 3.54230377e-03 3.87359837e-03
 7.72171254e-03 4.63812436e-03 3.72069317e-03 3.72069317e-03
 3.77166157e-03 3.10907238e-03 8.07849134e-03 4.35779817e-03
 7.44138634e-03 4.61264016e-03 5.07135576e-03 4.17940877e-03
 4.63812436e-03 5.27522936e-03 4.25586137e-03 5.24974516e-03
 4.43425076e-03 4.76554536e-03 5.45361876e-03 8.35881753e-03
 4.12844037e-03 5.50458716e-03 5.04587156e-03 4.43425076e-03
 6.65137615e-03 5.45361876e-03 6.77879715e-03 4.86748216e-03
 7.84913354e-03 5.81039755e-03 6.88073394e-03 9.19979613e-03
 7.72171254e-03 8.23139653e-03 8.23139653e-03 1.48572885e-02
 1.64118247e-02 1.81702345e-02 1.39143731e-02 1.34811417e-02
 1.39398573e-02 6.77879715e-03 1.74311927e-02 2.10244648e-02
 2.54841998e-05]
thetaNeg = [5.04216548e-03 8.84590435e-04 1.13522439e-02 5.89726956e-04
 4.86524739e-03 2.77171670e-03 1.00253583e-03 2.41788052e-03
 1.38585835e-03 1.71020817e-03 1.88712626e-03 2.18198974e-03
 1.73969452e-03 1.85763991e-03 1.76918087e-03 1.50380374e-03
 1.35637200e-03 1.62174913e-03 2.18198974e-03 1.20894026e-03
 5.33702896e-03 2.24096243e-03 1.79866722e-03 1.09099487e-03
 2.15250339e-03 3.18452556e-03 2.94863478e-05 2.06404435e-03
 2.38839417e-03 1.41534470e-03 2.03455800e-03 2.91914843e-03
 4.39346583e-03 3.62682078e-03 1.85763991e-03 3.24349826e-03
 1.71020817e-03 4.21654774e-03 4.98319278e-03 3.77425252e-03
 2.35890783e-03 2.47685322e-03 3.47938904e-03 2.27044878e-03
 2.09353070e-03 2.29993513e-03 2.18198974e-03 2.68325765e-03
 2.32942148e-03 2.12301704e-03 3.71527983e-03 5.92675591e-03
 3.00760748e-03 9.19974052e-03 3.18452556e-03 3.42041635e-03
 9.67152209e-03 4.15757504e-03 4.54089756e-03 2.77171670e-03
 3.65630713e-03 4.39346583e-03 6.98826443e-03 3.03709383e-03
 6.78186000e-03 2.41788052e-03 3.74476617e-03 4.71781565e-03
 3.50887539e-03 5.42548800e-03 4.54089756e-03 5.57291974e-03
 3.33195730e-03 5.10113817e-03 3.21401191e-03 8.99333609e-03
 5.57291974e-03 6.92929174e-03 4.77678835e-03 4.01014330e-03
 3.86271156e-03 5.48446069e-03 1.08804623e-02 8.63949991e-03
 9.70100843e-03 6.81134635e-03 7.60747774e-03 6.84083269e-03
 8.28566374e-03 8.46258182e-03 1.15881347e-02 1.03202217e-02
 8.93436339e-03 2.06404435e-02 9.52409035e-03 1.14407030e-02
 1.53623872e-02 2.04635254e-02 1.71020817e-02 2.36185646e-02
 2.94863478e-05]
```

(Q6) You are required to provide the following function (and module) for grading
`yPredict,Accuracy =`
`naiveBayesMulFeaturetest(Xtest,ytest,thetaPos,thetaNeg)`
Please refer Code

(Q7) Use ”sklearnnaive bayes.MultinomialNB” from the scikit learn package to perform training and testing. Compare the results with your MNBC. Add the resulting Accuracy into the writeup.

Choice-1:

```
      MNBC classification accuracy = 77.33333333333333

      Sklearn MultinomialNB accuracy = 77.33333333333333
```

Choice-2:

```
      MNBC classification accuracy = 76.33333333333333

      Sklearn MultinomialNB accuracy = 76.33333333333333
```

Compared to custom MNBC implementation, the accuracy of SKlearn MNBC has the same accuracy for both Choice-1 and Choice-2.

(Q10) You are required to provide the following function (and module) for grading:

```
thetaPosTrue,thetaNegTrue = naiveBayesBernFeature
train(Xtrain,ytrain)
```

Please refer code

(Q11) Provide the resulting parameter estimations into the writing.

Choice-1 Thetas:

```
thetaPosTrue = [0.05128205 0.07692308 0.40740741 0.04131054 0.03846154 0.11680912
 0.10683761 0.14814815 0.0982906  0.12820513 0.08404558 0.12108262
 0.08262108 0.08974359 0.08547009 0.12393162 0.08974359 0.12535613
 0.08547009 0.12108262 0.24074074 0.1011396  0.0968661  0.12535613
 0.11111111 0.11823362 0.0014245  0.15527066 0.11396011 0.17378917
 0.12250712 0.07122507 0.14387464 0.14957265 0.11538462 0.05982906
 0.11396011 0.1965812  0.03988604 0.12535613 0.13390313 0.07834758
 0.16524217 0.14672365 0.14245014 0.14245014 0.15669516 0.14957265
 0.11396011 0.16809117 0.13247863 0.18091168 0.26495726 0.1011396
 0.15811966 0.17094017 0.31766382 0.19088319 0.16239316 0.17236467
 0.16096866 0.14102564 0.32193732 0.15669516 0.3019943  0.1951567
 0.1951567  0.19088319 0.2037037  0.22792023 0.18376068 0.23361823
 0.19230769 0.21794872 0.22364672 0.35042735 0.18233618 0.23646724
 0.19088319 0.15099715 0.20940171 0.20512821 0.3048433  0.21937322
 0.25071225 0.21652422 0.26923077 0.32051282 0.3048433  0.30769231
 0.30911681 0.49287749 0.47720798 0.56267806 0.45726496 0.42450142
 0.45584046 0.26210826 0.54273504 0.56552707 0.0014245 ]
thetaNegTrue = [0.18518519 0.03988604 0.34615385 0.02706553 0.2022792  0.11396011
 0.04558405 0.0997151  0.05982906 0.06125356 0.08262108 0.09259259
 0.07407407 0.07977208 0.07834758 0.06837607 0.05698006 0.06552707
 0.08831909 0.05698006 0.20512821 0.09401709 0.08119658 0.05128205
 0.0968661  0.13105413 0.0014245  0.08689459 0.0982906  0.06267806
 0.08404558 0.05555556 0.16239316 0.15527066 0.07407407 0.12393162
 0.07834758 0.16239316 0.18091168 0.1025641  0.08404558 0.08119658
 0.14672365 0.0954416  0.08689459 0.1011396  0.09259259 0.11253561
 0.08831909 0.09259259 0.11680912 0.23789174 0.11680912 0.29344729
 0.12393162 0.14529915 0.33903134 0.15811966 0.16951567 0.12393162
 0.15099715 0.16809117 0.25641026 0.11253561 0.25071225 0.0997151
 0.13390313 0.1965812  0.14529915 0.18945869 0.18518519 0.21225071
 0.14102564 0.18233618 0.12820513 0.32193732 0.20512821 0.24786325
 0.16381766 0.13390313 0.12535613 0.16524217 0.36324786 0.32193732
 0.26780627 0.2037037  0.26353276 0.23504274 0.29059829 0.28917379
 0.33760684 0.35042735 0.29487179 0.54273504 0.31766382 0.35185185
 0.45868946 0.48860399 0.49430199 0.57692308 0.0014245 ]
```

(Q12) You are required to provide the following function (and module) for grading:

```
yPredict,Accuracy =
naiveBayesBernFeaturetest(Xtest,ytest,thetaPosTrue,thetaNegTrue)
```

Please refer code

Add the resulting Accuracy into the writing.

Choice-1:

```
    BNBC classification accuracy = 61.5
```

Choice-2:

```
    BNBC classification accuracy = 71.0
```

2. Sample QA Questions

Question 1. Bayes Classifier

Suppose you are given the following set of data with three Boolean input variables a, b, and c, and a single Boolean output variable G.

a	b	c	G
1	0	1	1
1	1	1	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	1
0	0	0	1
0	0	1	0

For item (a), assume we are using a naive Bayes classifier to predict the value of G from the values of the other variables.

(a) According to the naive Bayes classifier, what is $P(G = 1|a = 1 \wedge b = 1)$?

1/3

P (G=1 | a=1 ∧ b=1) = P (G=1 ∧ a=1 ∧ b=1) / P (a=1 ∧ b=1)

= P (G=1) * P(a=1 ^ b=1|G =1) / P(a=1 ^ b=1)

= P (G=1) * P(a=1|G=1) * P(b=1|G =1) / P(a=1) * P (b=1)

= (4/8 * 2/4 * 1/4) / (4/8 + 3/8) = 1/3

Please provide a one-sentence justification for the following TRUE/FALSE questions.

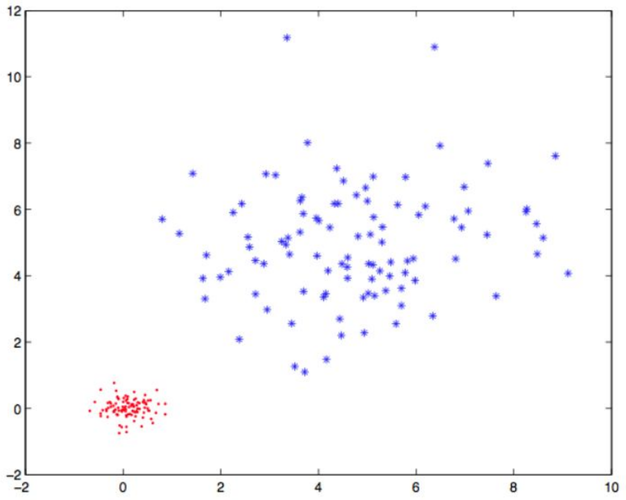
(b) (**True/False**) Naive Bayes Classifier and logistic regression both directly model $p(C|X)$.

False
BC -> estimates parameters for *P*(*c*) and *P*(*X*|*c*). Generative model.
Logistic Regression -> directly estimates the parameters of *P*(*C*|*X*). Discriminative model.

(c) (**True/False**) Gaussian Naive Bayes Classifier and Gaussian Mixture Model are similar since both assume that $p(X|\text{cluster} == i)$ follows Gaussian distribution.

True: Both models assume gaussian distribution but GMM represents as the weighted sum of multiple Gaussian distributions whereas Gaussian Naive Bayes represents single Gaussian distribution.

(d) (**True/False**) when you train Gaussian Naive Bayes Classifier on data samples provided in the following Figure, using separate covariance for each class, $\Sigma_1 \neq \Sigma_2$, the decision boundary will be linear. Please provide a one-sentence justification.



False. Gaussian Bayes classifier with Σ1 != Σ2 will give quadratic decision boundary. When Σ1 != Σ2, it is QDA formulation hence quadratic boundary.