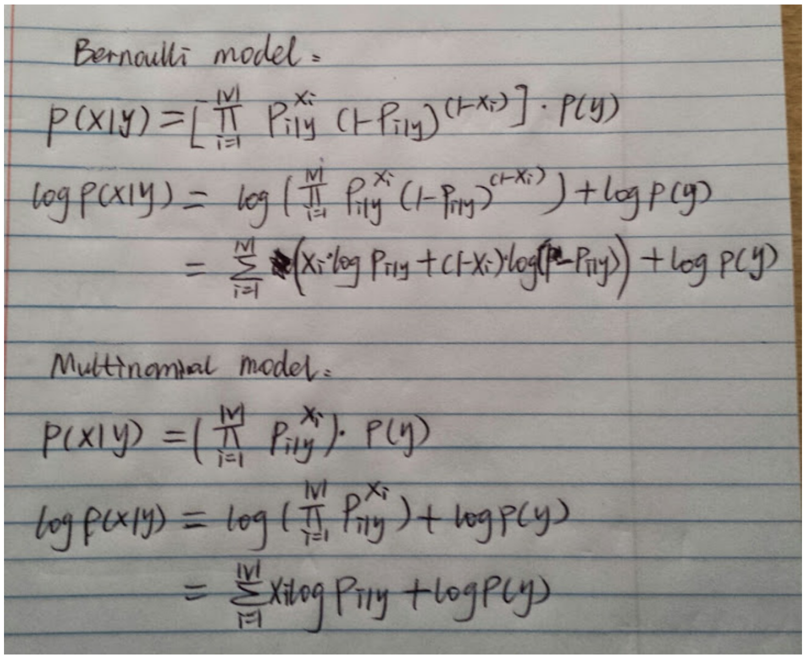
CS534 Project 2
Yaonan Zhong
April 30, 2014

1. Using the log of probability to perform classification



Bernoulli model:

$$P(X|y) = \left[\prod_{i=1}^{|V|} P_{i|y}^{X_i} (1-P_{i|y})^{(1-X_i)}\right] \cdot P(y)$$

$$\log P(X|y) = \log\left(\prod_{i=1}^{|V|} P_{i|y}^{X_i} (1-P_{i|y})^{(1-X_i)}\right) + \log P(y)$$

$$= \sum_{i=1}^{|V|} \left(X_i \cdot \log P_{i|y} + (1-X_i)\log(1-P_{i|y})\right) + \log P(y)$$

Multinomial model:

$$P(X|y) = \left(\prod_{i=1}^{|V|} P_{i|y}^{X_i}\right) \cdot P(y)$$

$$\log P(X|y) = \log\left(\prod_{i=1}^{|V|} P_{i|y}^{X_i}\right) + \log P(y)$$

$$= \sum_{i=1}^{|V|} X_i \log P_{i|y} + \log P(y)$$

Fig. 1 Using the log of probability to avoid underflow issues

2. Overall testing accuracy

Bernoulli model = 64%, Multinomial model = 78%. We can see that multinomial model is more accurate than bernoulli model, since more information are captured by multinomial model.

3. Confusion matrix

```
 107    0    0    0    0    1    0    0    0    0    1    7    0    4    3  159    3   31    2    0
   0  152    4   18    2   47    0    1    1    0    0  130    1    4    8   17    1    3    0    0
   0    8  125   47    1   38    0    1    0    1    1  129    1    3    8   23    2    2    1    0
   0    2   13  241    6    4    3    3    0    0    1   90   14    3    3    5    3    1    0    0
   0    5    2   51  142    4    4    3    0    1    0  112    8    8    6   17    7   11    2    0
   0   11    3    7    0  280    0    0    1    0    0   68    0    2    5    7    1    4    1    0
   0    4    4   48   12    2  155   29    5    1    2   49   13    3   13   13   10   15    4    0
   1    1    0    0    0    0    2  298    4    1    0   15    1    3    6   15   34   12    2    0
   0    0    0    0    0    0    0   14  313    0    0   11    0    3    1   13   20   22    0    0
   0    0    1    0    1    1    0    1    0  280   19   22    0    3    0   34    9   24    2    0
   0    0    0    0    0    1    0    0    0    2  374    3    0    2    0    7    4    6    0    0
   0    3    0    0    0    0    0    0    0    1    0  378    0    0    1    5    3    4    0    0
   0    5    0    9    2    1    1    7    2    0    0  148  140   14   11   20   12   21    0    0
   0    4    0    2    0    0    2    1    0    0    0    6    2  288    3   60    8   16    1    0
   0    1    0    0    0    1    0    0    0    0    1   20    3    4  317   17    4   22    2    0
   1    0    0    0    1    0    0    0    0    0    0    0    0    0    1  391    2    2    0    0
   0    0    0    0    0    0    1    0    1    1    0    9    0    0    0    6  322   22    1    1
   1    0    0    0    1    0    0    1    0    1    1    3    0    0    0    7    4  356    1    0
   0    0    0    0    1    0    0    0    0    0    0    8    0    1    7   22   97   42  132    0
  12    1    0    0    0    0    0    0    0    0    0    5    0    1    5  166   25   16    3   17
```

Fig. 2 Bernoulli confusion matrix

```
235    0    0    0    0    1    0    0    0    0    1    1    1    2    3   45    3   10    7    9
  3  296    6   12    7   22    1    3    2    0    0   17    4    4    7    4    0    0    1    0
  3   33  207   58   11   31    0    2    2    2    1   17    1    4    4    5    0    0    9    1
  0    8   15  305   21    2    4    6    0    0    1    6   23    0    1    0    0    0    0    0
  0    8   10   37  273    3    4    4    1    1    0    6   17    8    2    0    3    0    6    0
  0   42    7   10    2  306    1    0    2    1    0   10    0    0    3    2    1    1    2    0
  0    8    4   50   20    1  226   33    5    0    1    3   11    2    3    4    2    3    6    0
  1    1    0    2    0    1    5  356    4    2    0    1    4    0    2    1    4    2    9    0
  0    1    0    0    0    0    0   26  353    2    0    1    1    1    0    1    4    2    5    0
  4    1    0    1    1    2    3    3    1  345   17    2    2    0    0    3    1    2    9    0
  2    0    0    0    0    0    1    1    0    4  381    1    0    2    1    2    0    1    3    0
  0    4    1    1    2    1    1    0    0    0    0  361    3    2    0    2    8    0    8    1
  2   18    0   27    8    3    1   10    2    0    0   46  259    6    3    6    0    2    0    0
 10    7    1    3    0    0    0    4    0    1    0    1    3  324    3   17    3    6   10    0
  3    7    0    0    0    2    0    0    1    0    1    4    4    4  335    5    1    2   22    1
  7    2    1    0    1    2    0    0    0    0    0    1    0    1    0  377    2    2    1    1
  1    0    0    0    1    0    1    2    1    1    1    3    0    1    2    3  325    2   16    4
 12    1    0    0    0    0    0    2    1    1    1    4    0    0    0    8    3  325   18    0
  6    1    0    0    0    1    0    1    0    0    0    3    0    3    7    3   95    5  184    1
 47    3    0    0    0    0    0    0    1    0    0    1    0    3    5   70   19    5    8   89
```

Fig. 3 Multinomial confusion matrix

The above confusion matrices show that, for Bernoulli model, documents with class (12, 16, 17, 18) are most confused, and for Multinomial model, document with class 2, 4 and 19 are most confused. And most elements are at the diagonals.

4. Priors and overfitting

Prediction accuracy against Dirichlet distribution(1+a,...,1+a) with different a as the prior for Multinomial model.

Alpha = [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1]

Accuracy = [0.78387742 0.78894071 0.79706862 0.80373085 0.80466356 0.78107928]

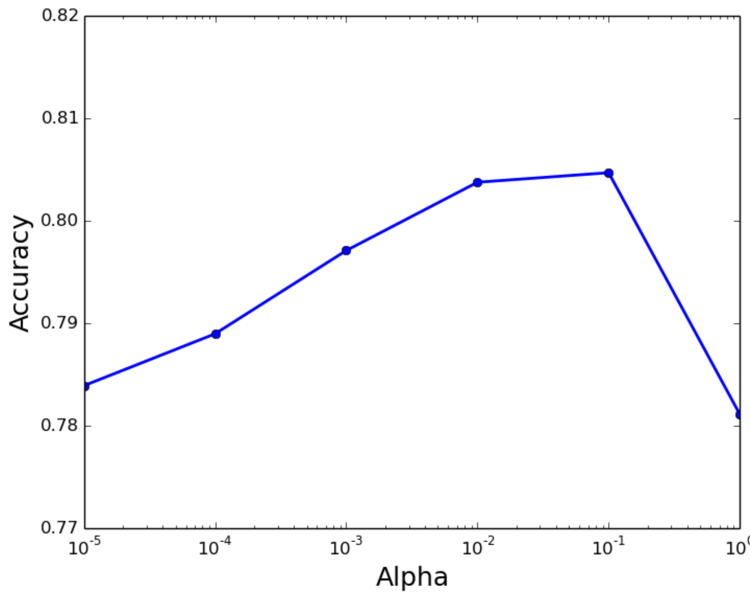
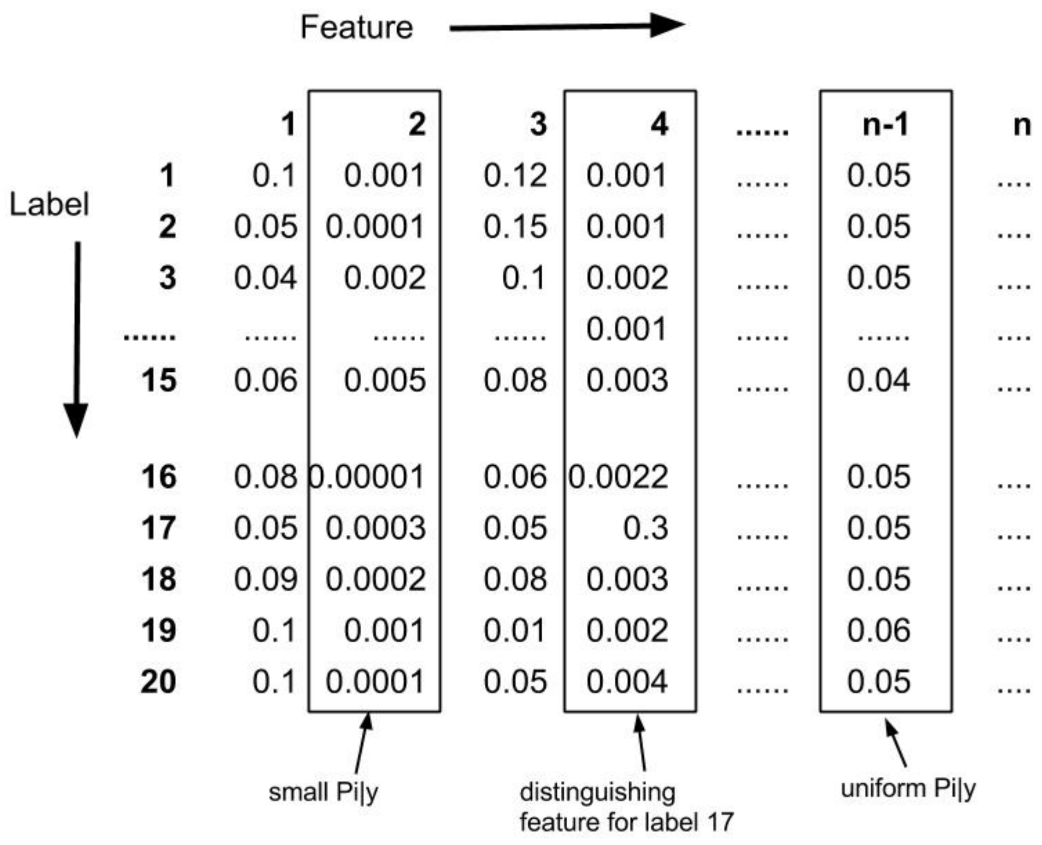


Fig. 4 Prediction accuracy against Alpha

Figure 4 shows that the prediction accuracy increases with alpha at the range of [1e-5, 0.1], and then decrease as alpha increases. We can consider extreme cases to explain this trend. If alpha approaches zero, the MAP will have no effect on our Bayes classifier, which becomes MLE estimation again. It If alpha is too large, the estimation will approach uniform probability, which also decrease our accuracy. A proper MAP estimation can avoid overfitting and extreme probability values.

5. Identifying important features

In order to identify important features, we can consider the contribution of each feature to the final prediction since they are conditional independence based on Naive Bayes assumption. And there are several ways to achieve our goal. The first method is to remove the feature i which has very small probability of Pi|y for all the labels, since it has little contribution to the prediction. The second is to remove the feature i which has almost uniform probability of Pi|y for all the labels, since we can not distinguish the label from this feature. And the last way is to keep those features with large Pi|y to a specific label, since we can distinguish this label with this feature. Figure 5 shows the above methods. Actually these methods are complementary.


Feature
Label
small Pi|y
distinguishing feature for label 17
uniform Pi|y

	1	2	3	4	......	n-1	n
1	0.1	0.001	0.12	0.001	......	0.05	....
2	0.05	0.0001	0.15	0.001	......	0.05	....
3	0.04	0.002	0.1	0.002	......	0.05	....
......	......	......	......	0.001	......	......	....
15	0.06	0.005	0.08	0.003	......	0.04	....
16	0.08	0.00001	0.06	0.0022	......	0.05	....
17	0.05	0.0003	0.05	0.3	......	0.05	....
18	0.09	0.0002	0.08	0.003	......	0.05	....
19	0.1	0.001	0.01	0.002	......	0.06	....
20	0.1	0.0001	0.05	0.004	......	0.05	....

Fig. 5 Identify important features and remove little-contribution features

The way to implement the above methods is computing the SSE(sum of squared error) or SD(standard deviation) of each feature. If the SSE or SD of a feature is very small and approach zero, it means that feature is likely to have uniform Pi|y or very small Pi|y, so we just remove this feature. If the SSE or SD is very large, it means that this feature has extreme Pi|y value for some particular labels.

Here I choose Multinomial model and Dirichlet distribution with Alpha=0.1, which gives us the highest accuracy in figure 4. And the standard deviation is computed. In this project, the size of vocabulary is 61188, and the SD distribution of these feature is in the range of [2.54e-7, 7.46e-3]. So I try to remove those features with SD<=1E-6, and the size of removed features is 8020, which is 13% of the original size.

Original features: Accuracy = 0.80466356
Remove 13% features: Accuracy = 0.8061292471685543

We can see that the new feature set has higher accuracy than the old one, which confirms our optimization methods. And we can explore more thresholds for SD, and remove different number of features.

Source code:

- learn.py,
- testBer.py,
- testMul.py,
- testMulAlp.py,
- featSD.py
- predictAcc.py
- predictAccRM.py
- featureVector.py