Project 2 Classification Analysis

Huang xinxin 104589081
Lu qiujing 704617222
Niu longjia 304590762

Problem (a) Description

To get started, **plot a histogram of the number of documents per topic to make sure they are evenly distributed**. Then report the number of documents in the Computer Technology and Recreational Activity.

Problem (a) Solution:

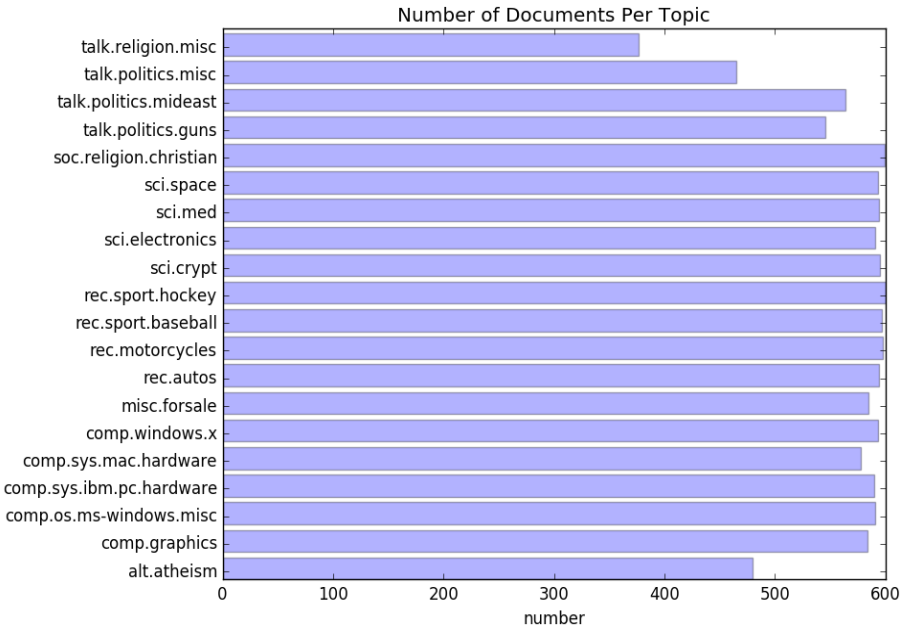


Fig a-1. Distribution of documents in each category

In the figure above, we can see that the number of documents in each category is almost **evenly distributed**. So we get a balanced dataset to training our model. The total number of documents in the two groups above, Computer Technology and Recreational Activity, are **1765** and **1789**, respectively.

Problem (b) Description:

1. Tokenize each document and extract all the words that appear in your documents, excluding the stop words, punctuations, and different stems of a word.
2. Create the TF-IDF vector representations. Report the final number of terms you extracted.

Problem (b) Solution:

This representation should be succinct as not to include too much irrelevant information, which leads to computational intractability and over fitting and at the same time capture essential features of the document.

1. Considering the Term Frequency-Inverse Document Frequency (TF-IDF) metric, we use the TfidfVectorizer to do the tokenization and TF-IDF transform.

Discussion: although the normalized TF-IDF is widely used, sometimes the binary occurrence marker can do better than features. Taking the short texts with noise input into consideration, the information extracted by the latter can be steadier.

2. In this project, we use the stem function in the NLTK.

Discussion: Although it is not as fast as pystem, considering the relatively small size of data set and convenience, it is still worth trying.

The size of original term-document matrix without preprocessing is **(4732,79218)**. By excluding stop words, punctuations and different stems of a word, the size reduces to **(4732, 70065)**. We can also use the min_df, max_df parameters to adjust the feature numbers. (For example, if min_df=0.1, it turns into**(4732, 71)**). It shows that the low frequency terms occupy the majority of data.

Problem (c) Description:

We define a new measure called TF*ICF with a similar definition except that a class sits in place of a document. The formula is shown below. Find the 10 most significant terms in the following classes (comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, misc.forsale, and soc.religion.christian), with respect to the above measure.

$$(0.5 + 0.5\frac{f_{t,c}}{\max_{t'} f_{t',c}}) \times \log\frac{|C|}{|c \in C : \ t \in c|}$$

Problem (c) Solution:

The formula of TF*ICF creates an ambiguity. There are two ways to interpret this formula. First, we consider c as a subset of all 20 classes, in this case, ICF may be calculated by logarithm of |C|=20 divided by |c| of 20 classes which has the specific term in it. Thus the value may vary from log(20/1) to log(20/20). Theoretically, since the other 16 classes may contribute significantly to the denominator of the logarithm ICF, it will lower the relevance of the terms in the selected 4 classes. The top 10 most significant terms for each of 4 classes are shown below in Fig c-1. As we can see, the terms extracted are somehow irrelevant in terms of its class.

comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	misc.forsale	soc.religion.christian
Rank 1: aspi4do	Rank 1: powerbook	Rank 1: sabretooth	Rank 1: clh
Rank 2: st412	Rank 2: lciii	Rank 2: liefeld	Rank 2: liturgi
Rank 3: st506	Rank 3: c650	Rank 3: hobgoblin	Rank 3: kulikauska
Rank 4: 1542b	Rank 4: adb	Rank 4: de7	Rank 4: mmalt
Rank 5: f300r	Rank 5: bmug	Rank 5: 02106	Rank 5: caralv
Rank 6: balog	Rank 6: iivx	Rank 6: uccxkvb	Rank 6: monophysit
Rank 7: penev	Rank 7: q700	Rank 7: radley	Rank 7: mussack
Rank 8: t560i	Rank 8: iifx	Rank 8: kou	Rank 8: sspx
Rank 9: scsiha	Rank 9: jartsu	Rank 9: koutd	Rank 9: schismat
Rank 10: husak	Rank 10: firstclass	Rank 10: snes	Rank 10: atterlep

Fig c-1. top 10 most significant terms by first understanding

The second understanding is that we consider c as a subset of 4 classes, in this case, ICF may be calculated by logarithm of |C|=20 divided by |c| of 4 classes which has the specific term in it. thus the value may vary from log(20/1) to log(20/4). Theoretically, this will enhance the relevance of the extracted terms and will process data much more faster than the first understanding. The top 10 most significant terms for each of 4 classes are shown below in Fig c-2. Take soc.religion.christian for example, terms such as "christ", "belief" and "cathol" are strongly related to religious category.

comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	misc.forsale	soc.religion.christian
Rank 1: cmos	Rank 1: powerbook	Rank 1: wolverin	Rank 1: christ
Rank 2: aspi4do	Rank 2: lciii	Rank 2: obo	Rank 2: belief
Rank 3: rri	Rank 3: c650	Rank 3: hiram	Rank 3: scriptur
Rank 4: dx2	Rank 4: pds	Rank 4: hulk	Rank 4: cathol
Rank 5: wlsmith	Rank 5: iisi	Rank 5: ticket	Rank 5: atho
Rank 6: exe	Rank 6: 040	Rank 6: deck	Rank 6: arrog
Rank 7: desonia	Rank 7: adb	Rank 7: sabretooth	Rank 7: jew
Rank 8: dcoleman	Rank 8: hade	Rank 8: liefeld	Rank 8: clh
Rank 9: st412	Rank 9: bmug	Rank 9: rider	Rank 9: jewish
Rank 10: st506	Rank 10: 68030	Rank 10: hobgoblin	Rank 10: revel

Fig c-2. top 10 most significant terms by second understanding

Problem (d) Description:

Apply LSI to the TF-IDF matrix and pick k=50; so each document is mapped to a 50-dimensional vector. Use the selected features in your learning algorithms.

Problem (d) Solution:

In this part, we implement SVD to the TF-IDF matrix of the training data and get a k*d size matrix U_k, the matrix of left singular column vectors corresponding to the largest k singular values. Then we map TF-IDF of training data into the k-dimension representation $D_k = U_k * (TFIDF)$. For the datas in test set, we also map the TF-IDF matrix into the same subspace, which means using the same projection matrix U_k as the training set.

Problem (e) Description:

Use SVM method to separate the documents into Computer Technology vs. Recreational Activity groups. In your submission, plot the **ROC curve**, report the **confusion matrix** and calculate the **accuracy**, **recall** and **precision** of your classifier

Problem (e) Solution:

(1). ROC Curve is shown in Fig e-1.

As shown in Fig e-1, this method has sharp ROC curve above the 45 degree axis, which means that the SVM classifier can achieve high true positive rate with tolerable false positive rate and the TPR can increase greatly with the small increase in FPR.

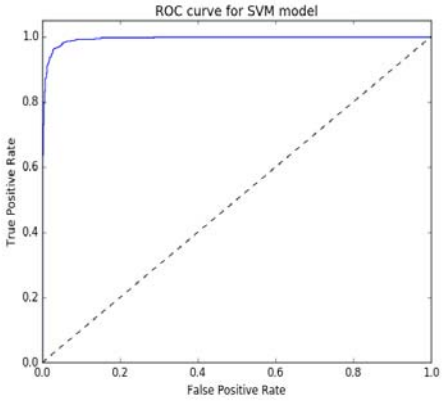


Fig e-1. ROC of hard margin SVM

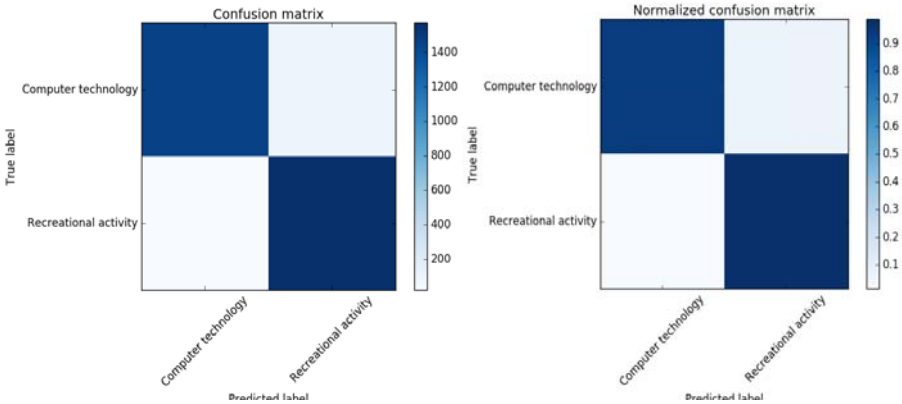


Fig e-2. Confusion Matrix of hard margin SVM

(2). Confusion Matrix

It is used to evaluate the quality of the output of a classifier on the test data set. The diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabeled by the classifier. The higher the diagonal values of the confusion matrix the better, indicating more correct predictions.

As shown in the confusion matrix in Fig e-2, the diagonal values of the confusion matrix are relatively high so the prediction of the classificatory is acceptable. Moreover, the lighter color of the left bottom than right upper ones means that the errors have different distribution: the probability of being misclassified into Recreational activity is higher than being misclassified into Computer Technology. But since the errors are both small, it can be viewed as balanced for the two classes. Detailed statistics are listed in Table e-1.

	Predicted Label = 0	Predicted Label = 1	
Actual label = 0	TN = 1461	FP = 99	Sum of actual 0 = 1560
Actual label = 1	FN = 21	TP = 1569	Sum of actual 1 =1590
	Sum of predicted 0 =1482	Sum of predicted 1 =1668	Sum = 3150

Table e-1. Confusion Matrix of hard margin SVM

(3). The accuracy, recall and precision

	precision	recall	f1-score	support
Computer technology	0.99	0.94	0.96	1560
Recreational activity	0.94	0.99	0.96	1590
avg / total	0.96	0.96	0.96	3150

Table e-2. Accuracy, precision and recall of hard margin SVM

Problem (f) Description:

Repeat the previous part with the soft margin SVM and, using a 5-fold cross-validation, find the best value of the parameter ɣ in the range {10^(-k)|−3≤k≤3,k∈Z}. Report the **confusion matrix** and calculate the **accuracy, recall and precision** of your classifier.

Problem (f) Solution:

In this part, we try to solve an optimal problem with object function

$$\min \frac{1}{2} \| w \|_2^2 + \gamma \sum_{i}^{n} \xi_i$$

In this function, a set of slack variables are introduced . Each slack variable shows the amount of error that the classifier makes on a given example. We can control the tradeoff between these two components by changing the parameter ɣ . In our experiment, we implemented two kind of kernels for the SVM classifier with ɣ in $\{10^{-k} | -3 < k < 3, k \in Z\}$. We use the 5 cross-validation to evaluate the model performance. We report the average accuracy of 5 tests with different parameters below in the Table f-1.

Kernel	0.001	0.01	0.1	1	10	100	1000
linear	0.5049	0.5051	0.9717	0.9730	0.9738	0.9738	0.9734
rbf	0.5049	0.5049	0.5049	0.9596	0.9727	0.9734	0.9751

Table f-1. Accuracy of soft margin with different parameters

As we can see from the table above, these two classifiers get almost perfect accuracy score when ɣ >= 1. So we implement with ɣ = 10 and linear kernel. The results are shown below:

(1). ROC Curve is shown in Fig f-1.

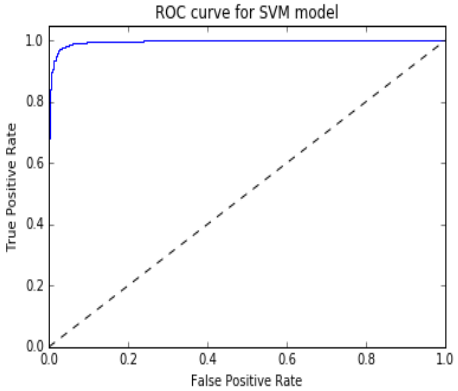


Fig f-1. ROC of soft margin SVM

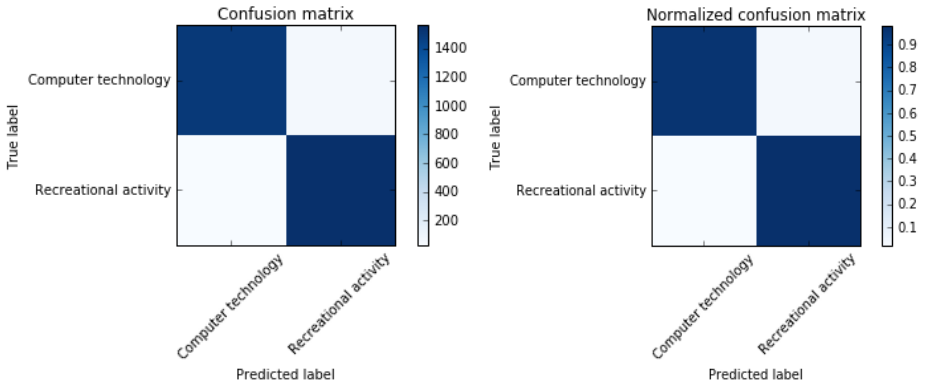


Fig f-2. Confusion matrix of soft margin SVM

(2). Confusion Matrix is shown in Fig f-2 and Table f-2.

	Predicted Label = 0	Predicted Label = 1	
Actual label = 0	TN = 1502	FP = 58	Sum of actual 0 = 1560
Actual label = 1	FN = 32	TP = 1558	Sum of actual 1 =1590
	Sum of predicted 0 =1534	Sum of predicted 1 =1616	Sum = 3150

Table f-2 Confusion Matrix of soft margin SVM

(3). The accuracy, recall and precision is shown in Table f-3

	precision	recall	f1_score	support
Computer technology	0.98	0.96	0.97	1560
Recreational activity	0.96	0.98	0.97	1590
avg / total	0.97	0.97	0.97	3150

Table f-3. Accuracy, recall and precision of soft margin SVM

Accuracy = 0.9714

Problem (g) Description:

Train a Gaussian naïve Bayes classifier and plot the ROC curve for different values of the threshold on class probabilities. You should report your **ROC curve** along with that of the other algorithms. Again, Report the **confusion matrix** and calculate the **accuracy, recall and precision** of your classifier.

Problem (g) Solution:

GaussianNB implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood.

(1). ROC Curve is shown in Fig g-1.

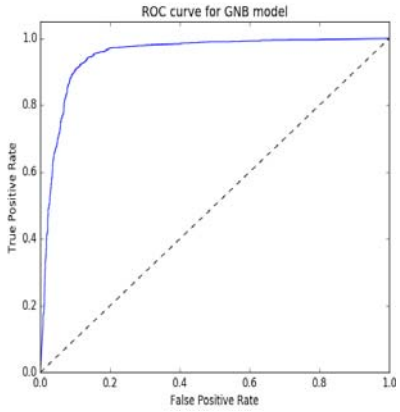


Fig g-1. ROC of Gaussian NB

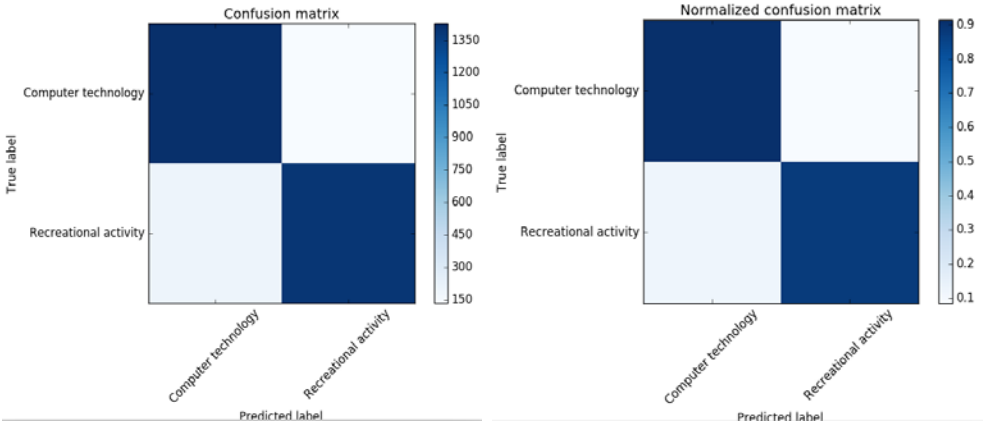


Fig g-2. Confusion matrix of Gaussian NB

(2). Confusion Matrix is shown in Fig g-2 and Table g-1.

	Predicted Label = 0	Predicted Label = 1	
Actual label = 0	TN = 1427	FP = 133	Sum of actual 0 = 1560
Actual label = 1	FN = 194	TP = 1396	Sum of actual 1 =1590
	Sum of predicted 0 =1621	Sum of predicted 1 =1529	Sum = 3150

Table g-1. Confusion Matrix of Gaussian NB

(3). The accuracy, recall and precision is shown in Table g-2.

	precision	recall	f1_score	support
Computer technology	0.88	0.91	0.90	1560
Recreational activity	0.91	0.88	0.90	1590
avg / total	0.90	0.90	0.90	3150

Table g-2. Accuracy, recall and precision of soft margin SVM

Accuracy = 0.8961

As shown above, although Naive Bayes is built on simple design and oversimplified assumptions by assuming that the value of a particular feature is independent of the value of any other feature, given the class variable. In the case of classification between computer technology and recreational activity, it achieves an accuracy of 89.6%, which is lower than 96% in SVM classification and 97% in soft margin SVM, possibly because the dependences between different features distribute unevenly among them. However, it is still competitive compared to more complicated SVM methods because its simplicity in data updating, modeling and coding.

Problem (h) Description:

Repeat the same task with the logistic regression classifier, and plot the ROC curve for different values of the threshold on class probabilities. You should report your ROC curve along with that of the other algorithms. Provide the same evaluation measures on this classifier

Problem (h) Solution:

Logistic regression, despite its name, is a linear model for classification rather than regression.
As an optimization problem, binary class L2 penalized logistic regression minimizes the following cost function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Similarly, L1 regularized logistic regression solves the following optimization problem

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

(1). ROC Curve is shown in Fig h-1.

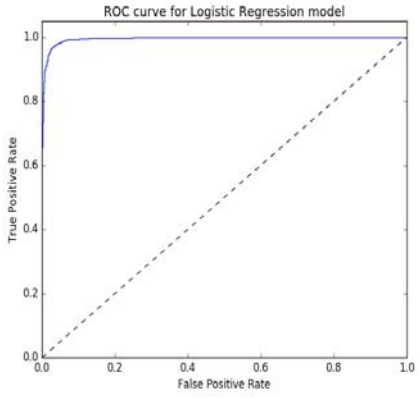


Fig h-1. ROC of Logistic Regression

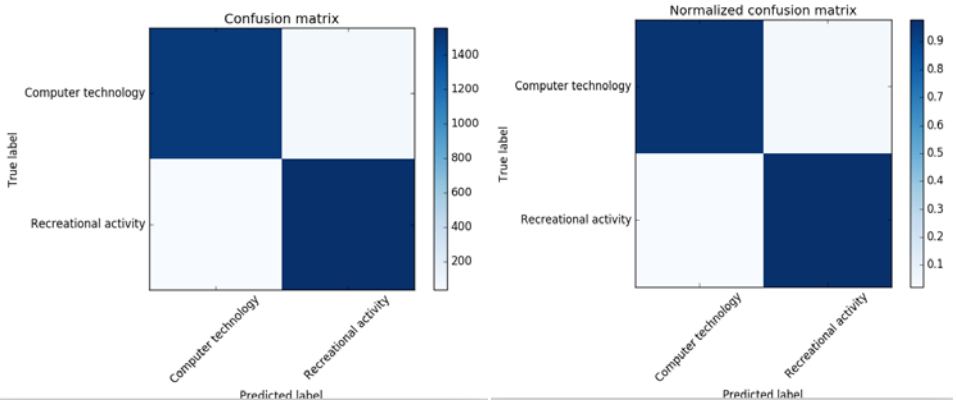


Fig h-2. Confusion matrix of Logistic Regression

(2). Confusion Matrix is shown in Fig h-2 and Table h-1.

	Predicted Label = 0	Predicted Label = 1	
Actual label = 0	TN = 1496	FP = 64	Sum of actual 0 = 1560
Actual label = 1	FN = 34	TP = 1556	Sum of actual 1 =1590
	Sum of predicted 0 =1621	Sum of predicted 1 =1529	Sum = 3150

Table h-1. Confusion Matrix of Logistic Regression

(3). The accuracy, recall and precision is shown in Table h-2.

	precision	recall	f1_score	support
Computer technology	0.98	0.96	0.97	1560
Recreational activity	0.96	0.98	0.97	1590
avg / total	0.97	0.97	0.97	3150

Table h-2. Accuracy, recall and precision of soft margin SVM

Accuracy = 0.9689

As shown from the results above, logistic regression classifier achieves 96.9% accuracy 97.9% recall and 96% precision which are nearly the same with SVM and soft margin SVM methods. Besides the high performance of the classifier, it is more robust because the independent variables don't have to be normally distributed, or have equal variance in each group. However, it also requires much more data to achieve stable, meaningful results, which does not happen in our case where each subclass consists of 500+ samples.

Problem (i) Description:

In this part, we aim to learn classifiers on the documents belonging to the classes mentioned in part b; namely comp.sys.ibm.pc.hardware , comp.sys.mac.hardware, misc.forsale, and soc.religion.christian. Perform Naïve Bayes classification and multiclass SVM classification (with both One VS One and One VS the rest method) and report the **confusion matrix** and calculate the **accuracy, recall and precision** of your classifiers.

Problem (i) Solution:

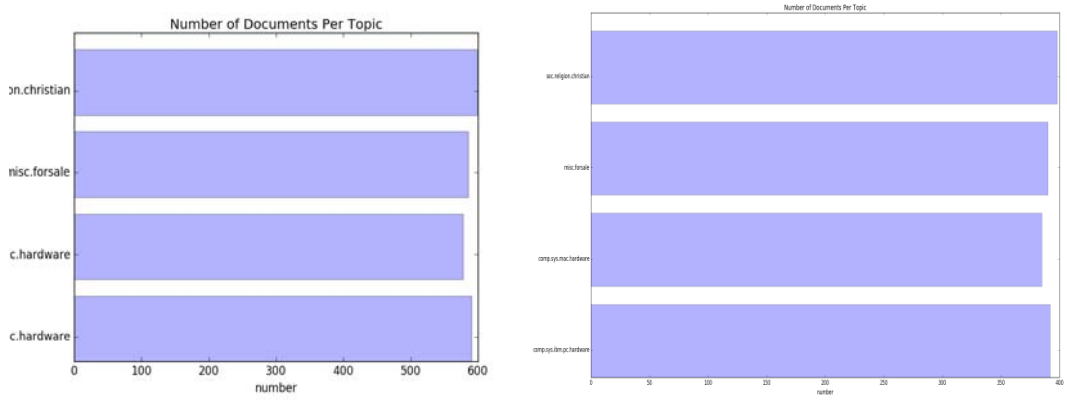


Fig i-1. Distribution of documents of different categories

Before the classification, we verify the training and testing dataset. They are balanced.

We implemented two type of bayes classifiers, Gaussian Naive Bayes and Multinomial Navie Bayes. Gaussian Naive Bayes has already mentioned in the previous part. Multinomial Naive Bayes is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). So we want to compare the performance with Gaussian one.

(1) Gaussian NB

(1.1). The accuracy, recall and precision of multi-class Gaussian NB is shown in Table i-1.

	precision	recall	f1-score	support
comp.sys.ibm.pc.hardware	0.63	0.61	0.62	392
comp.sys.mac.hardware	0.65	0.37	0.47	385
misc.forsale	0.51	0.81	0.63	390
soc.religion.christian	0.99	0.88	0.93	398
avg / total	0.70	0.67	0.66	1565

Table i-1. Accuracy, recall and precision of multi-class GaussianNB

Accuracy = 0.6677

(1.2). Confusion Matrix of multi-class Gaussian NB is shown in Fig i-2 and Table i-2.

	Predicted Label = 0	Predicted Label = 1	Predicted Label = 2	Predicted Label = 3	
Actual label = 0	238	39	114	1	Sum_a0 = 392
Actual label = 1	98	142	144	1	Sum_a1 =385
Actual label = 2	38	38	314	0	Sum_a2 =390
Actual label = 3	5	0	42	351	Sum_a3 =398
	Sum _p0 =379	Sum_p1 =219	Sum_p2 =614	Sum_p3 =353	Sum = 1565

Table i-2. Confusion matrix of multi-class GaussianNB

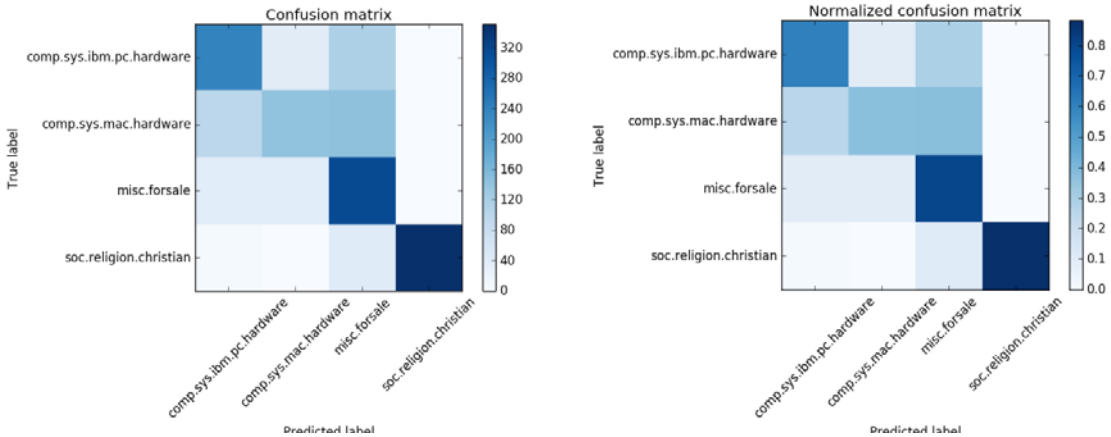


Fig i-2. Confusion matrix of multi-class Gaussian Naive Bayes

(2.1). The accuracy, recall and precision of multi-class Multinomial NB is shown in Table i-3.

	precision	recall	f1-score	support
comp.sys.ibm.pc.hardware	0.63	0.82	0.71	392
comp.sys.mac.hardware	0.89	0.54	0.67	385
misc.forsale	0.67	0.83	0.90	390
soc.religion.christian	0.99	0.83	0.90	398
avg / total	0.79	0.76	0.76	1565

Table i-3. Accuracy, recall and precision of multi-class Multinomial NB

Accuracy = 0.7565

(2.2). Confusion Matrix of multi-class Multinomial NB is shown in Fig i-3 and Table i-4.

	Predicted Label = 0	Predicted Label = 1	Predicted Label = 2	Predicted Label = 3	
Actual label = 0	323	16	53	0	Sum_a0 = 392
Actual label = 1	117	206	59	3	Sum_a1 =385
Actual label = 2	56	9	323	2	Sum_a2 =390
Actual label = 3	18	0	48	332	Sum_a3 =398
	Sum_p0 =514	Sum_p1 =231	Sum_p2 =483	Sum_p3 =337	Sum = 1565

Table i-4. Confusion matrix of multi-class Multinomial NB

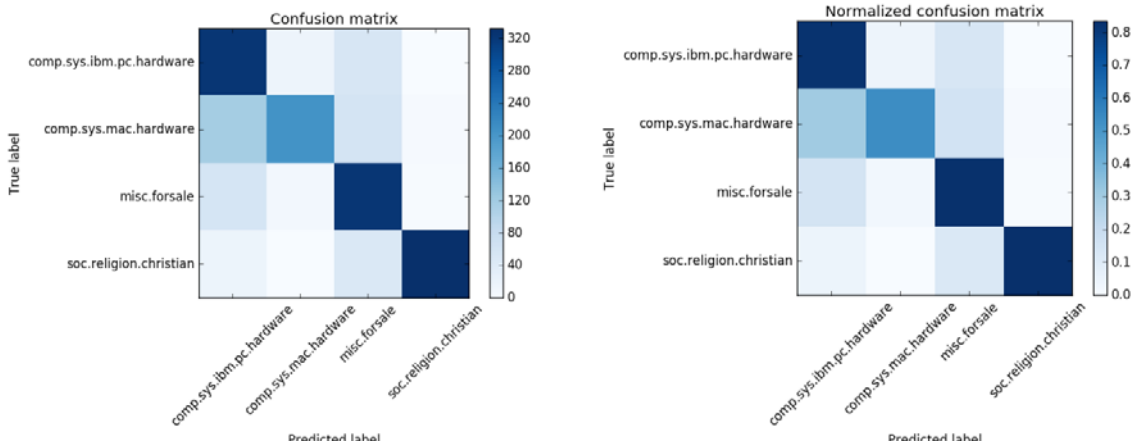


Fig i-3. Confusion matrix Multinomial Naive Bayes

It shows that both models can do a good classification work.

(3) SVM ,one-vs-the-rest

(3.1). The accuracy, recall and precision of multi-class OVR SVM is shown in Table i-5.

one advantage of this approach is its interpretability. Since each class is represented by one and one classifier only, it is possible to gain knowledge about the class by inspecting its corresponding classifier.

	precision	recall	f1-score	support
comp.sys.ibm.pc.hardware	0.82	0.73	0.77	392
comp.sys.mac.hardware	0.84	0.76	0.80	385
misc.forsale	0.83	0.89	0.86	390
soc.religion.christian	0.88	1.00	0.94	398
avg / total	0.84	0.85	0.84	1565

Table i-5. Accuracy, recall and precision of multi-class OVR SVM

(3.2). Confusion Matrix of multi-class OVR SVM is shown in Fig i-4:

[286 44 40 22]
[41 292 32 20]
[21 11 348 10]
[0 0 1 397]

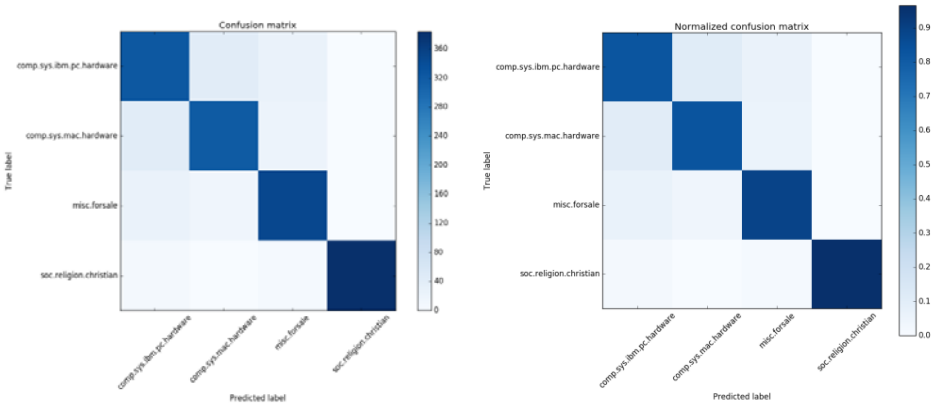


Fig i-4 Confusion Matrix of one-vs-rest SVM

(4) SVM , one-vs-one

(4.1). The accuracy, recall and precision of multi-class OVO SVM is shown in Table i-6.

This method is usually slower than one-vs-the-rest, due to its square complexity. For example in this problem, it requires to fit 4*3/2=6 classifiers. However, since each of its learning parts involving only a small subset of

data, this method is suitable for the kernel problem because of its weakness in scalability. In the one-vs-the-rest, the complete dataset is used 4 times.

	precision	recall	f1-score	support
comp.sys.ibm.pc.hardware	0.81	0.83	0.82	392
comp.sys.mac.hardware	0.84	0.83	0.84	385
misc.forsale	0.87	0.89	0.88	390
soc.religion.christian	0.99	0.96	0.98	398
avg / total	0.88	0.88	0.88	1565

Table i-6. Accuracy, recall and precision of multi-class OVO SVM

(4.2). Confusion Matrix of multi-class OVO SVM is shown in Fig i-5:

[325 43 24 0]
[42 320 22 1]
[25 16 348 1]
[8 1 5 384]

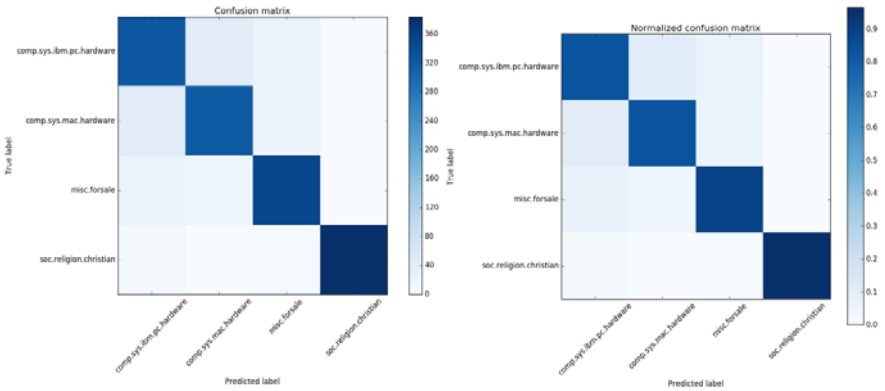


Fig i-5. Confusion Matrix of one-vs-one SVM

Comparing these two methods, the One VS One classifier causes less confusion in the classification because of its multiple verification and vote, but it has increasing complexity especially when the classes are large.

Discussion:

(1).classifier:

Output-code based strategies are fairly different from one-vs-the-rest and one-vs-one. Setting the parameter code size greater than 1, it requires more classifiers than one-vs-the-rest classifier but has similar accuracy as one-vs-one classifier. The error-correcting output codes have a similar effect to bagging.

with code size=3

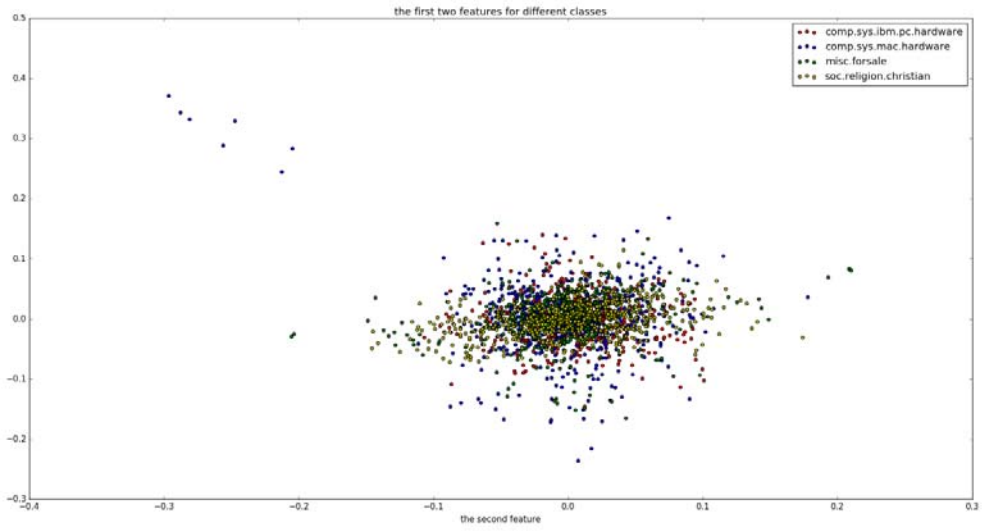
	precision	recall	f1-score	support
comp.sys.ibm.pc.hardware	0.86	0.78	0.81	392
comp.sys.mac.hardware	0.78	0.89	0.83	385
misc.forsale	0.91	0.89	0.90	390
soc.religion.christian	0.99	0.97	0.98	398
avg / total	0.88	0.88	0.88	1565

The confusion matrix is:

```
[304  70  18   0]
[ 26 341  17   1]
[ 21  22 346   1]
[  4   5   1 388]
```

There are also many other methods for training classifier for the large training dataset, such as bagging, boosting and stacking.

(2). SVM interpretation.



As shown in the picture, it is hard to interpret the classifier from low dimension, the kernel trick makes it possible to separate the data from higher dimensions. The consistent relatively low accuracy for the comp.sys.ibm.pc.hardware class may be due to the scattered points and similar pattern with comp.sys.mac.hardware.