ECE 219 Project1

Jiuru Shao
UID:204288539
Haoxiang Zhang
UID:104278461

Jan 28th 2018

1 Introduction

Classification is is the task of predicting a category that test data belong to, after training with data set with predefined categories. Classification is an essential area of data analysis. In this project, we research different methods for classifying textual data.

We firstly model text data by tokenizing each document into words, and then creating TFIDF vector representations. After that, we perform dimension reduction by applying both Latent Semantic Indexing (LSI) and Non-Negative Matrix Factorization (NMF) methods.

For learning algorithms, we study Support Vector Machine (SVM) classifier, Naive Bayes algorithm, logistic regression classifier under different document frequencies and dimension reduction methods. We plot the ROC curve, the confusion matrix, and calculate the accuracy, recall and precision of all learning algorithms to make comparisons.

Besides binary classifications, we also perform Naive Bayes multiclass classification, one versus one multiclass SVM classification and one versus rest multiclass SVM classification. Again, we perform the same evaluation metrics for these multiclass classification methods.

2 Part (a)

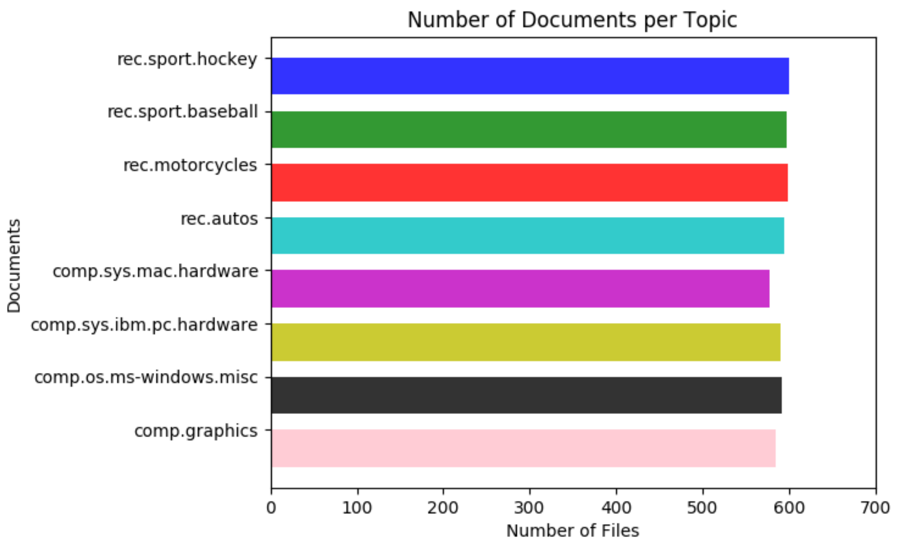


Figure 1: **Histogram of the number of training documents per class**

To make sure the number of documents in each class is evenly distributed, we plot a histogram of the number of document in each topic. The result shows that the data set of training documents is indeed balanced so we do not need to balance it any more.

3 Part (b)

For this part, we first load all the documents, then tokenize each document into words by creating a countVectorizer with our own defined tokenizing function. We use the package from nltk for tokenization and stem words removal. We then create the TFxIDF vector using the TfidfTransformer and fit the model with the docuements. When min_df = 2, the number of term is 12826, and when min_df is 5, the number of term is 5759.

Min DF	Number of Terms
2	12826
5	5759

Table 1: **Number of Terms**

4 Part (c)

In this part, we need to calculate the 10 most significant terms for the four classes, **comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, misc.forsale, soc.religion.christian.** Since there is not available library to use, we have to implement it ourselves. Instead of loading 4 classes at a time, we load one at a time, so that each classes will be filtered with min_df first. After fitting the countVectorizer, we merge the vocabulary of each class after removing all the duplicates. The number of term at this step is term. This way, we ensure all the term will have positive term frequency. Then we iterate through all the feature names of the 4 classes to get term frequencies(4 x #term matrix), and iterate all the 20 classes to get class frequency(1 x #term matrix), and multiply TF and ICF to get the final matrix. Then we get the top 10 elements of each row in the numpy array and return the result.

comp.sys.ibm. pc.hardware	comp.sys. mac.hardware	misc.forsale	soc.religion .christian
disk	nubus	new	faith
subject	organ	post	subject
line	duo	com	christ
use	centri	forsal	church
com	subject	dos	bibl
mb	simm	organ	atho
edu	line	subject	edu
drive	quadra	sale	jesus
ide	mac	line	christian
scsi	edu	edu	god

Table 2: **Top 10 Features of Each Class when Min_df is 2**

comp.sys.ibm. pc.hardware	comp.sys. mac.hardware	misc.forsale	soc.religion .christian
use	nubus	post	sin
jumper	lc	com	faith
disk	line	ship	atho
card	appl	forsal	edu
com	quadra	dos	christ
edu	duo	organ	bibl
mb	simm	subject	church
drive	scsi	line	jesus
ide	mac	sale	christian
scsi	edu	edu	god

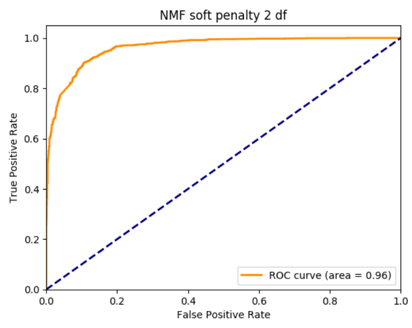
Table 3: **Top 10 Features of Each Class when Min_df is 5**

These terms make sense, and different min_df values create a difference in their ranks. But the selection of words are almost the same.

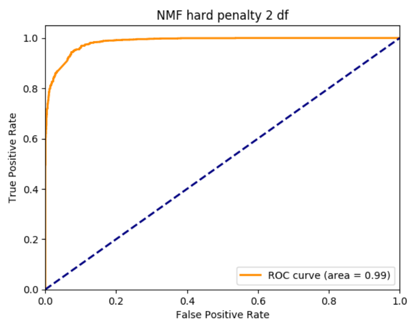
5 Part (d)

Once we have the TF x IDF matrix, we will need to perform dimension reduction on it by 1) LSI 2) NMF. We use the TruncatedSVD() and NMF modules in sklearn and set n_components=50, and we load the data for further use.

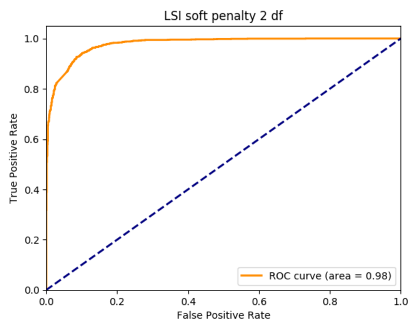
6 Part (e)



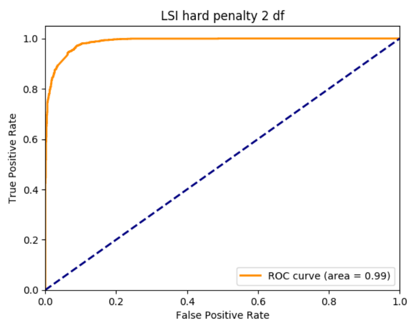
(a) NMF, Soft, df2



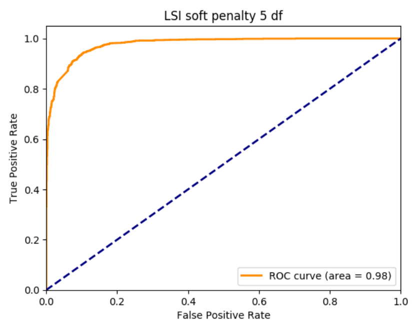
(b) NMF, Hard, df2



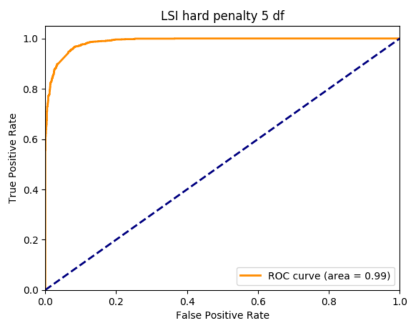
(c) LSI, Soft, df2



(d) LSI, Hard, df2

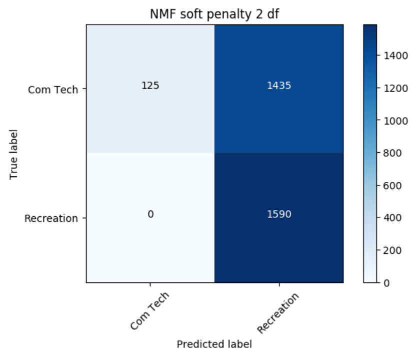


(e) LSI, Soft, df5

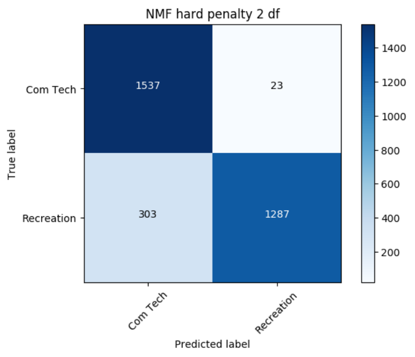


(f) LSI, Hard, df5

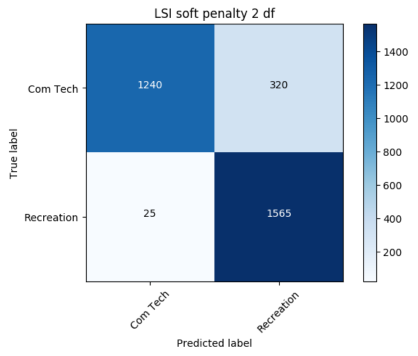
Figure 2: **ROC Curve for Part (e)**



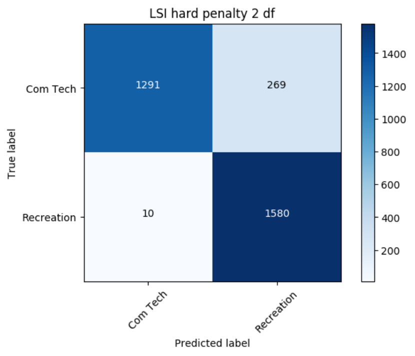
(a) NMF, Soft, df2



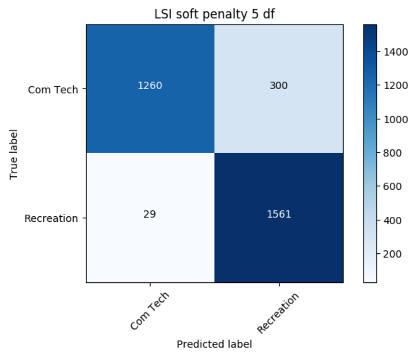
(b) NMF, Hard, df2



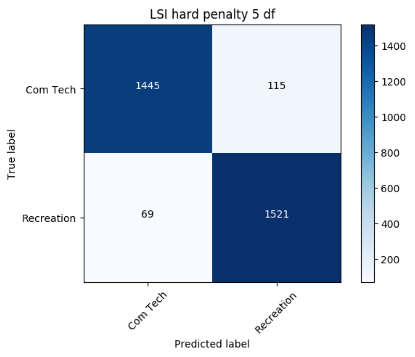
(c) LSI, Soft, df2



(d) LSI, Hard, df2

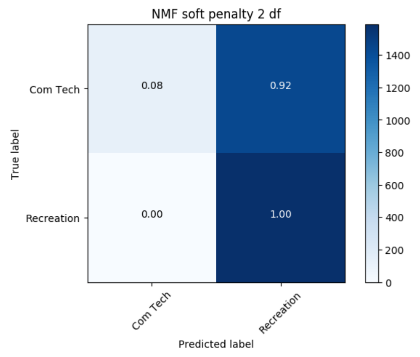


(e) LSI, Soft, df5

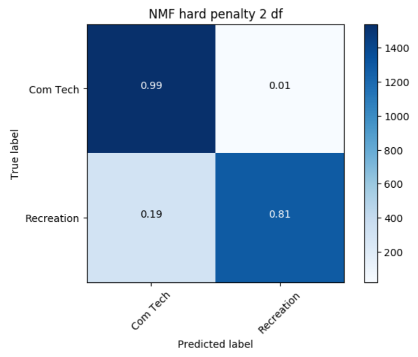


(f) LSI, Hard, df5

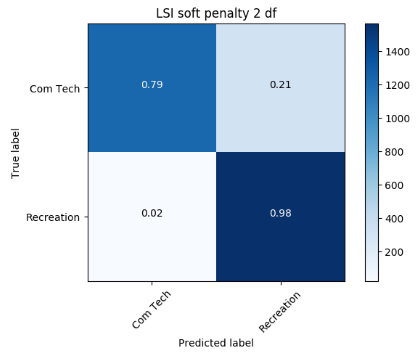
Figure 3: **Confusion Matrix for Part (e)**



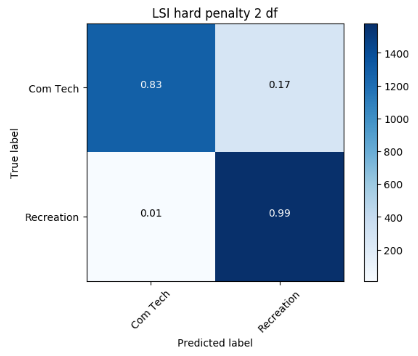
(a) NMF, Soft, df2



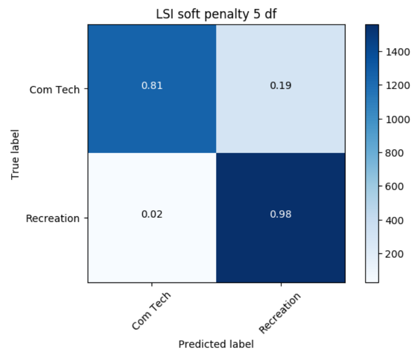
(b) NMF, Hard, df2



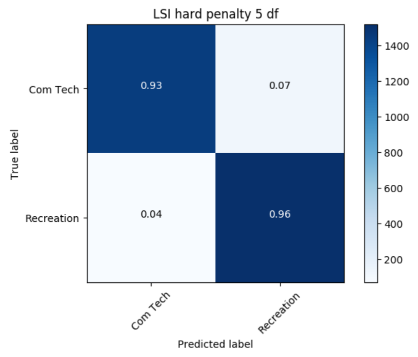
(c) LSI, Soft, df2



(d) LSI, Hard, df2



(e) LSI, Soft, df5



(f) LSI, Hard, df5

Figure 4: **Normalized Confusion Matrix for Part (e)**

Metrics	LSI,df2	LSI, df5	NMF,df2
Accuracy	0.911	0.942	0.897
Recall	0.994	0.957	0.809
Preicision	0.855	0.930	0.982

Table 4: **Hard Margin SVM Classifier**

Metrics	LSI,df2	LSI, df5	NMF,df2
Accuracy	0.890	0.896	0.544
Recall	0.984	0.982	1.0
Preicision	0.830	0.839	0.526

Table 5: **Soft Margin SVM Classifier**

Here are some of our finds. When min_df = 2, hard margin performs better than soft margin for both LSI and NMF. In fact, soft margin SVM using NMF data set makes no sense here: it appears all the examples in the test set are predicted to positive, causing a 1.0 recall.

After comparing different ROC Curve plots in Figure 2, different confusion matrix plots in Figure 3, different metrics (accuracy, recall, precision) in Table 4 and Table 5, we make following conclusions.

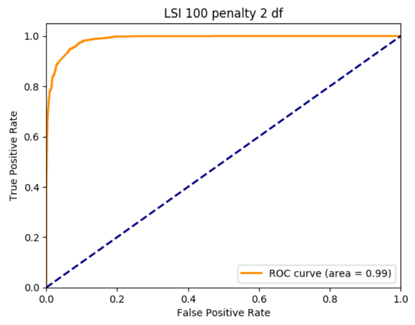
Hard margin SVM has better performance than soft margin SVM. LSI dimension reduction method has better performance than NMF method. Min_df = 5 has better performance than min_df = 2, so it is not worth keeping so many "rare" terms (i.e. min_df = 2 method is not worth trying).

Also, for a straight forward visualization of confusion matrix, we make plots of normalized confusion matrix as shown in Figure 4.

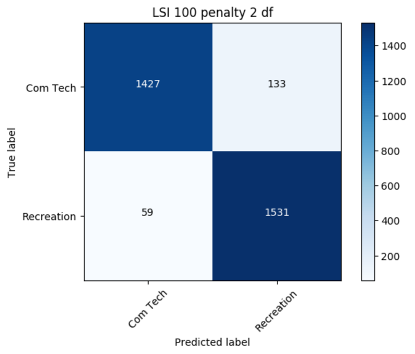
7 Part (f)

Method	Doc Freq	Best Penalty Paramter (γ)
LSI	2	100
LSI	5	10
NMF	2	100

Table 6: **Best SVM Parameter**

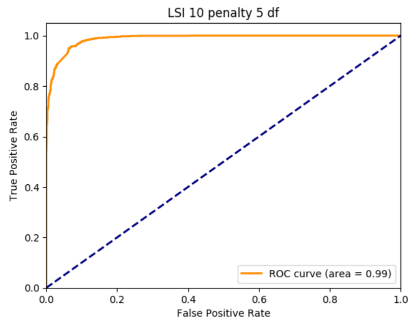


(a) ROC Curve

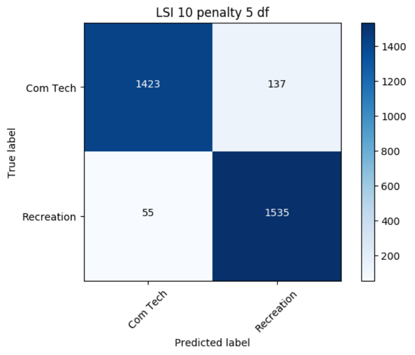


(b) Confusion Matrix

Figure 5: **LSI, penalty 100, df 2**

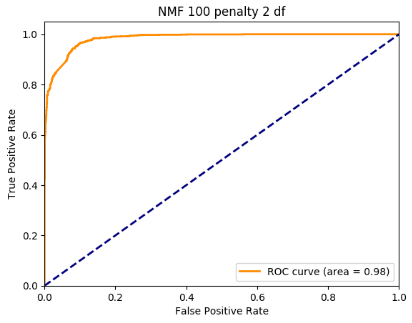


(a) ROC Curve

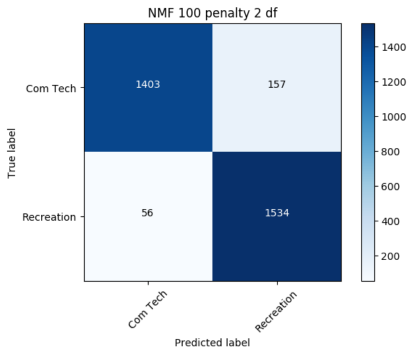


(b) Confusion Matrix

Figure 6: **LSI, penalty 10, df 5**



(a) ROC Curve



(b) Confusion Matrix

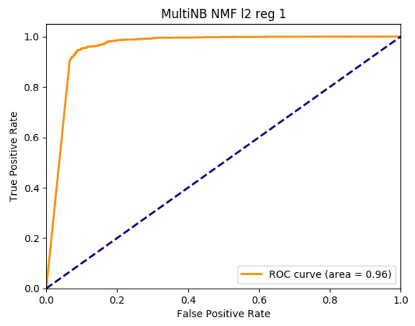
Figure 7: **NMF, penalty 100, df 2**

Metrics	LSI, penalty 100, df 2	LSI, penalty 10, df5	NMF, penalty 100, df2
Accuracy	0.939	0.939	0.932
Recall	0.963	0.965	0.965
Preicision	0.920	0.918	0.907

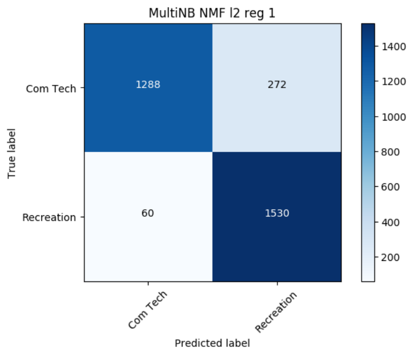
Table 7: **Best SVM Classifier Metrics**

After applying 5-fold cross validation, we find the best value of the penalty parameter γ for LSI method at document frequency 2 and 5, and for NMF method at document frequency 2. Best penalty parameters are listed in Table 6. We then perform same evaluations for each best paramters. ROC curve and confusion matrix plots are displayed from Figure 5 to Figure 7. Accuracy, recall and precision are listed in Table 7. From these figures and tables, it is very clear that each SVM classifier with best penalty parameter achieves state of the art performance.

8 Part (g)



(a) ROC Curve



(b) Confusion Matrix

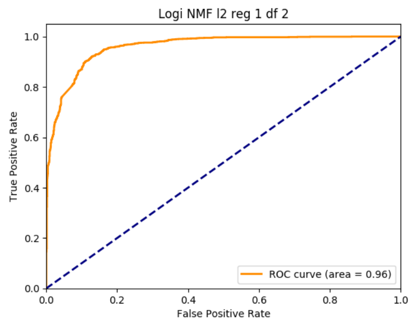
Figure 8: **Naive Bayes, NMF, df 2**

Metrics	NMF, Naive Bayes, df 2
Accuracy	0.909
Recall	0.964
Preicision	0.870

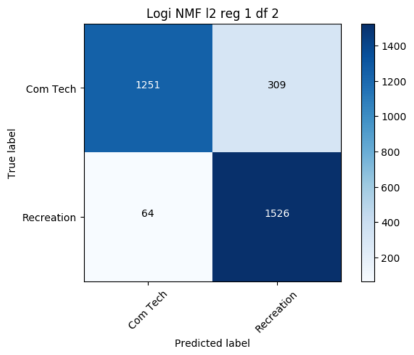
Table 8: **Naive Bayes Metrics**

For Naive Bayes algorithm, we only do the experiment with NMF method at document frequency of 2. Because this algorithm cannot tolerate negative values which LSI method. From Figure 8 and Table 8, Naive Bayes is not a bad choice for binary classification task. But it is still not as good as SVM classifier when comparing with previous SVM metrics.

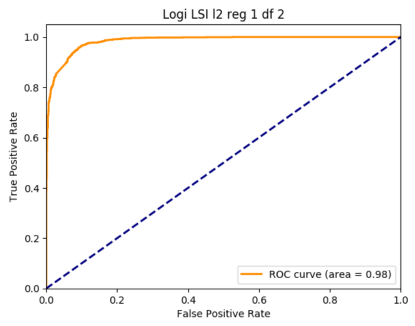
9 Part (h)



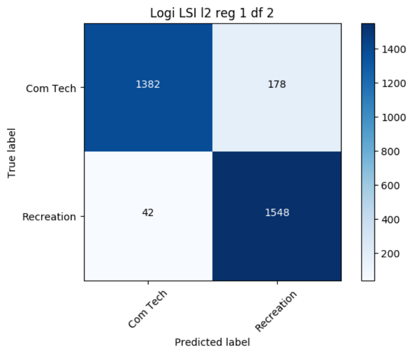
(a) NMF, Default Logi Regr, df2



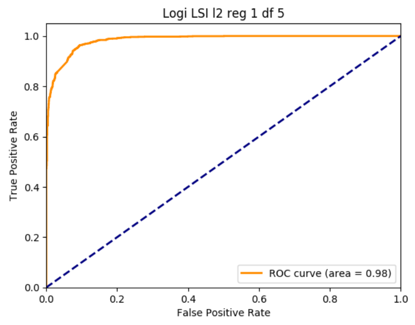
(b) NMF, Default Logi Regr, df2



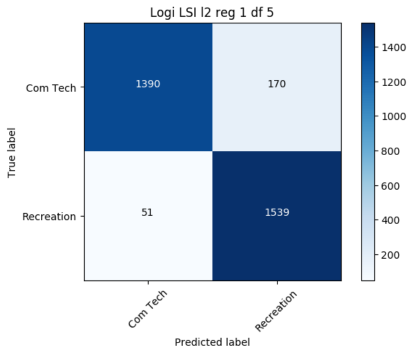
(c) LSI, Default Logi Regr, df2



(d) LSI, Default Logi Regr, df2



(e) LSI, Default Logi Regr, df5



(f) LSI, Default Logi Regr, df5

Figure 9: **ROC Curve and Confusion Matrix**

From Figure 9 and Table 9, we can conclude that in logistic regression classifier, LSI dimension reduction method has better performance than NMF

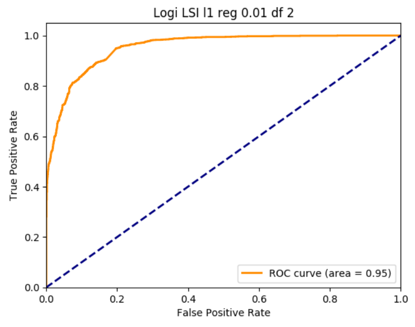
Metrics	NMF, Logistic, df 2	LSI, Logistic, df2	LSI, Logistic, df5
Accuracy	0.893	0.930	0.930
Recall	0.964	0.972	0.969
Preicision	0.846	0.898	0.899

Table 9: **Logistic Regression Classifier Metrics**

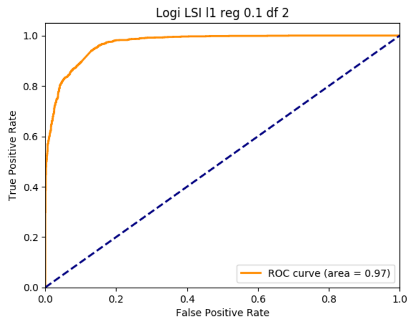
method. Besides, LSI method has similar performance at document frequency of 2 and 5. Also, logistic regression classifier with LSI method has similar performance as best suppport vector machine classifier has in part f.

10 Part (i)

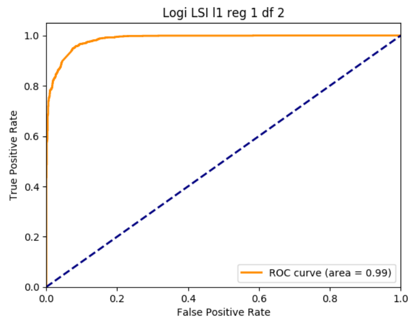
For this part, we try logistic regression for LSI dimension reduction method at document frequency of 2 and 5, and NMF method at document frequency of 2. For each method, we try both l_1 and l_2 norm regularizations, and sweep through $\{10^k | -2 \leq k \leq 3, k \in Z\}$ regularization coefficients. As a result, we have 3 $method \times 2\ norm \times 6\ reg\ coef = 36$ ROC Curve plots and 36 Confusion Matrix plots. We firstly present these figures and list related evaluation metrics, and then make analysis.



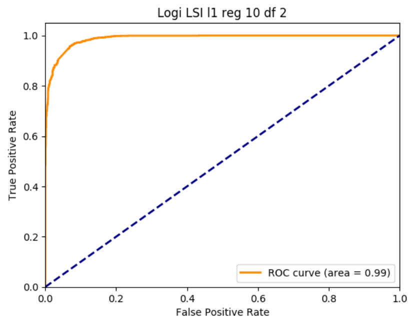
(a) LSI, l1, reg 0.01, df 2



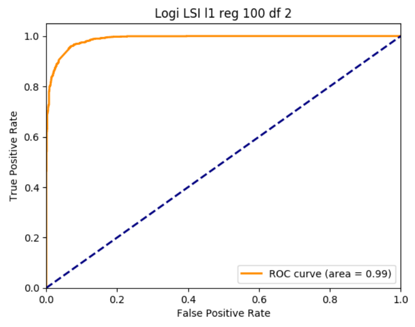
(b) LSI, l1, reg 0.1, df 2



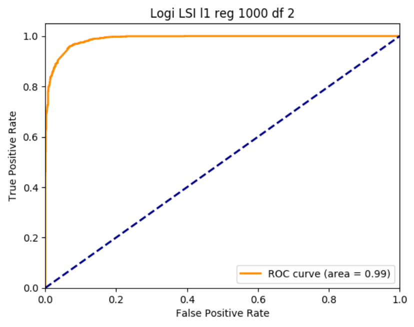
(c) LSI, l1, reg 1, df 2



(d) LSI, l1, reg 10, df 2

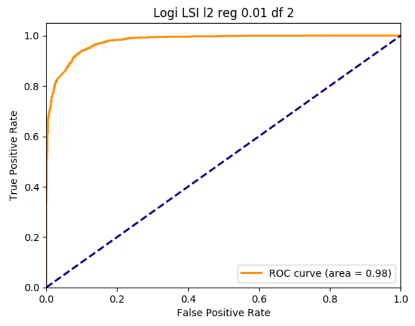


(e) LSI, l1, reg 100, df 2

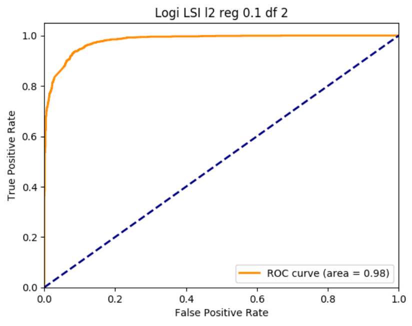


(f) LSI, l1, reg 1000, df 2

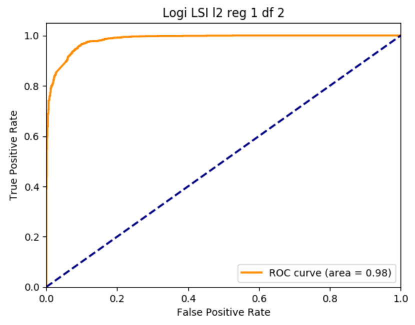
Figure 10: **ROC Curves of LSI with l1 penalty and df 2 of Part (i)**



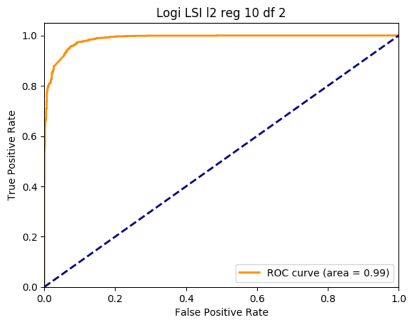
(a) LSI, l2, reg 0.01, df 2



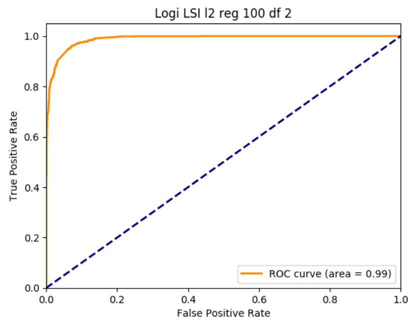
(b) LSI, l2, reg 0.1, df 2



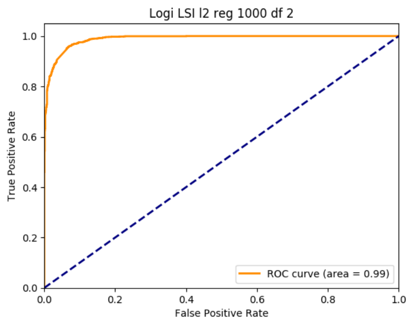
(c) LSI, l2, reg 1, df 2



(d) LSI, l2, reg 10, df 2

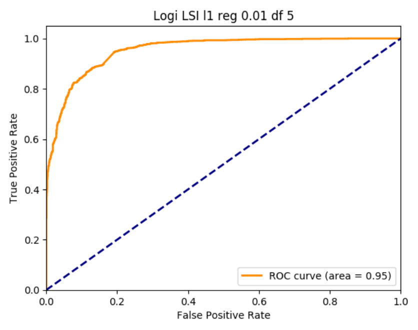


(e) LSI, l2, reg 100, df 2

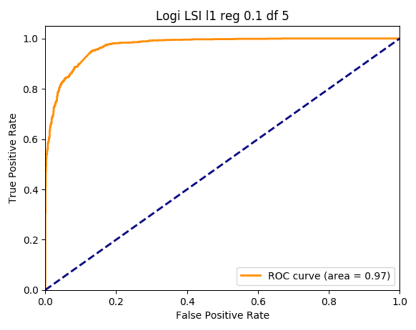


(f) LSI, l2, reg 1000, df 2

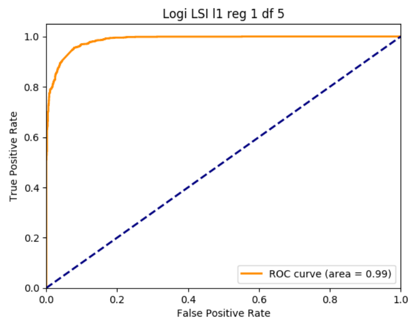
Figure 11: **ROC Curves of LSI with l2 penalty and df 2 of Part (i)**



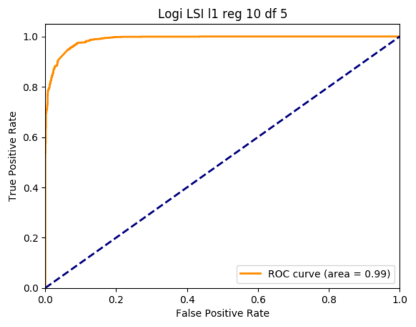
(a) LSI, l1, reg 0.01, df 5



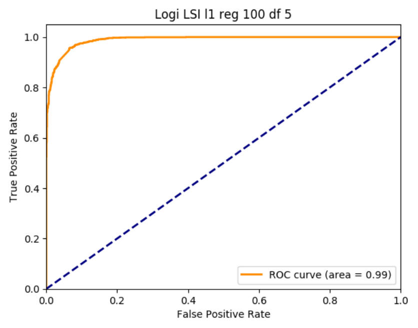
(b) LSI, l1, reg 0.1, df 5



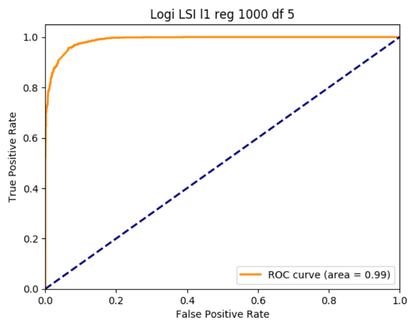
(c) LSI, l1, reg 1, df 5



(d) LSI, l1, reg 10, df 5

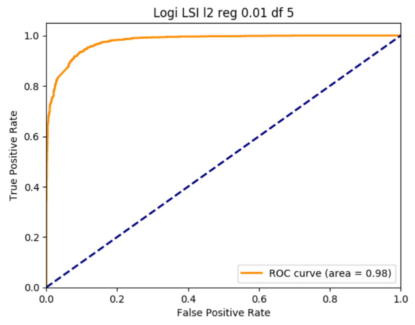


(e) LSI, l1, reg 100, df 5

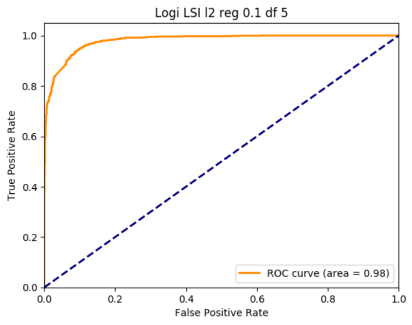


(f) LSI, l1, reg 1000, df 5

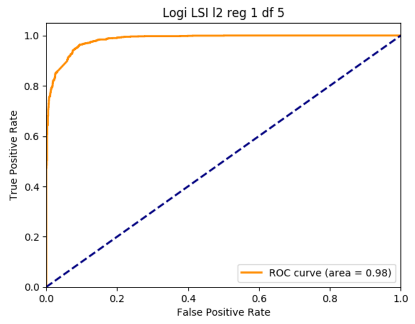
Figure 12: **ROC Curves of LSI with l1 penalty and df 5 of Part (i)**



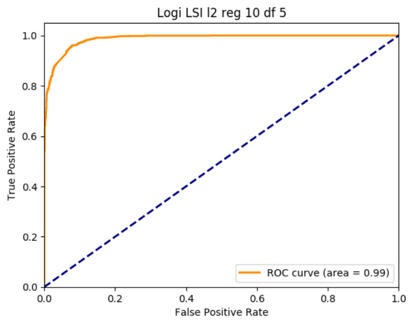
(a) LSI, l2, reg 0.01, df 5



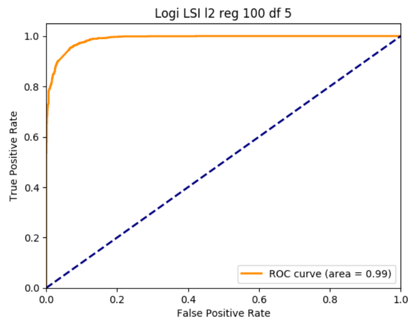
(b) LSI, l2, reg 0.1, df 5



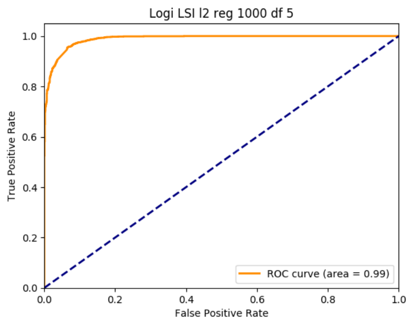
(c) LSI, l2, reg 1, df 5



(d) LSI, l2, reg 10, df 5

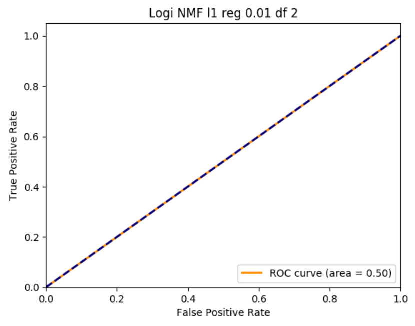


(e) LSI, l2, reg 100, df 5

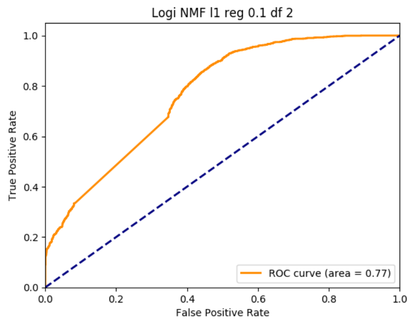


(f) LSI, l2, reg 1000, df 5

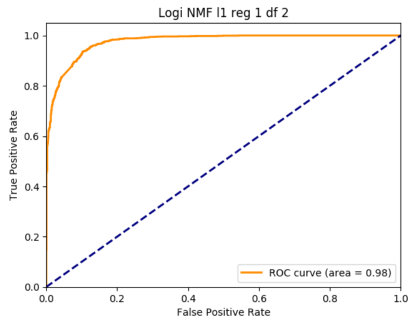
Figure 13: **ROC Curves of LSI with l2 penalty and df 5 of Part (i)**



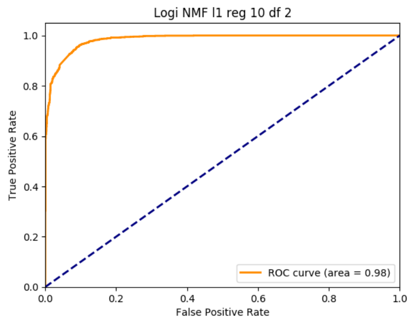
(a) NMF, l1, reg 0.01, df 2



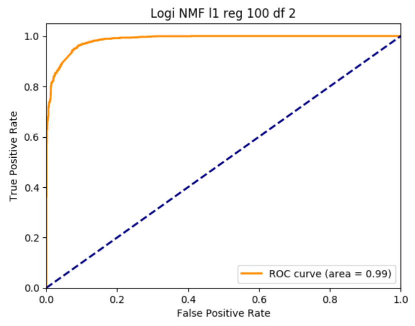
(b) NMF, l1, reg 0.1, df 2



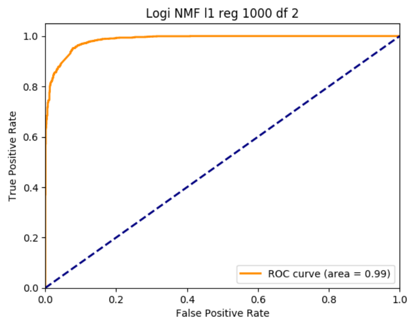
(c) NMF, l1, reg 1, df 2



(d) NMF, l1, reg 10, df 2

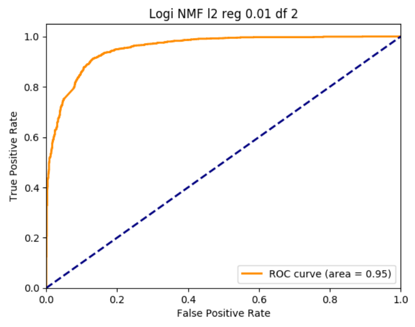


(e) NMF, l1, reg 100, df 2

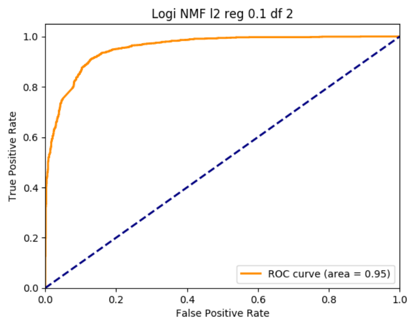


(f) NMF, l1, reg 1000, df 2

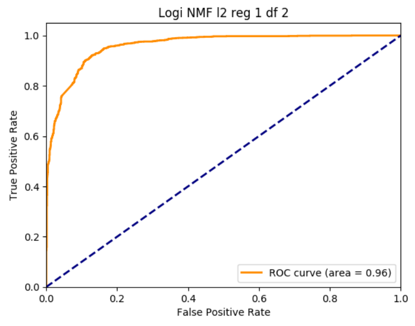
Figure 14: **ROC Curves of NMF with l1 penalty and df 2 of Part (i)**



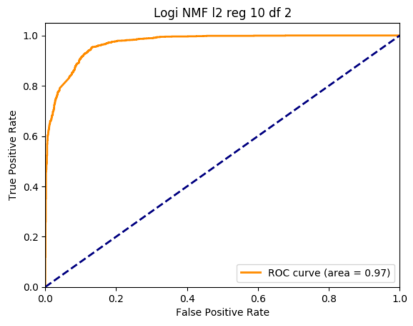
(a) NMF, l2, reg 0.01, df 2



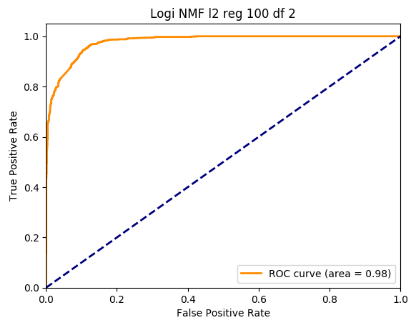
(b) NMF, l2, reg 0.1, df 2



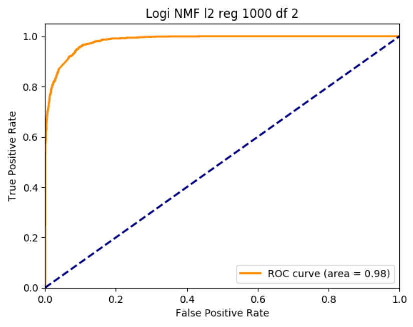
(c) NMF, l2, reg 1, df 2



(d) NMF, l2, reg 10, df 2

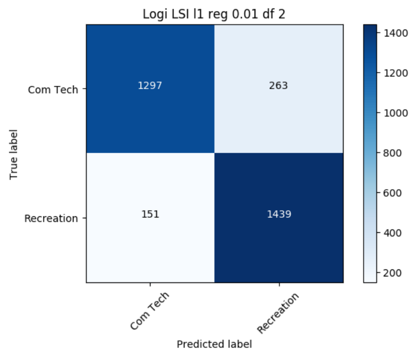


(e) NMF, l2, reg 100, df 2

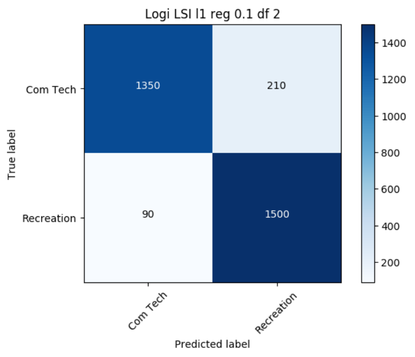


(f) NMF, l2, reg 1000, df 2

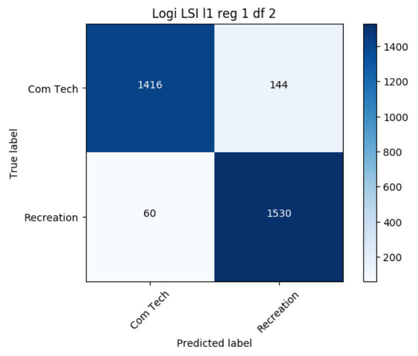
Figure 15: **ROC Curves of NMF with l2 penalty and df 2 of Part (i)**



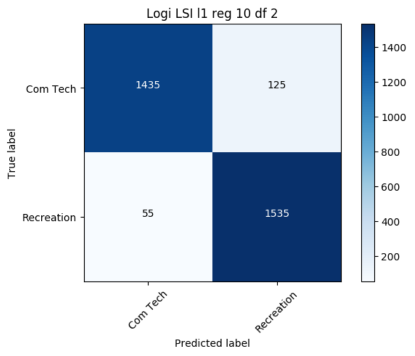
(a) LSI, l1, reg 0.01, df 2



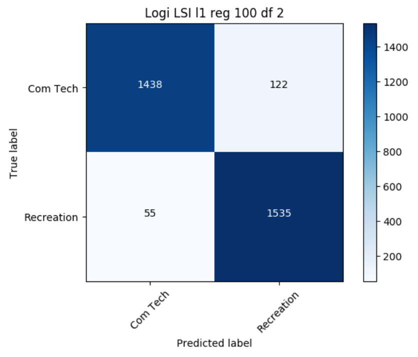
(b) LSI, l1, reg 0.1, df 2



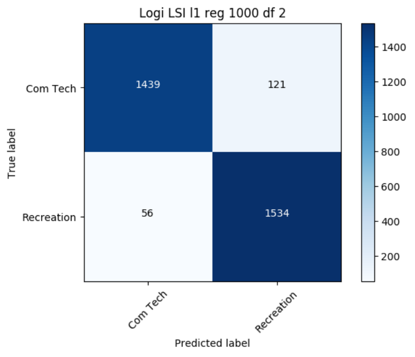
(c) LSI, l1, reg 1, df 2



(d) LSI, l1, reg 10, df 2

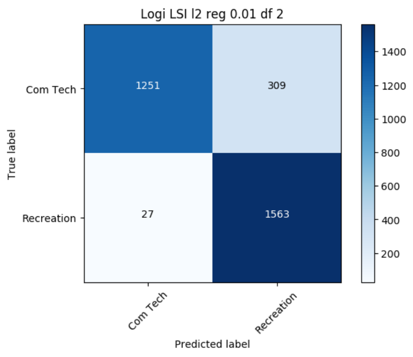


(e) LSI, l1, reg 100, df 2

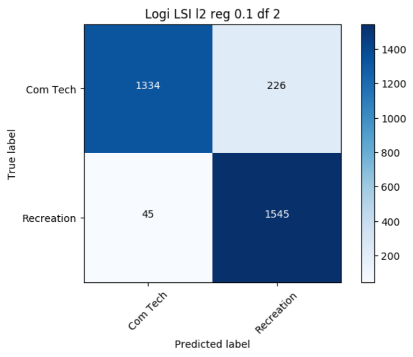


(f) LSI, l1, reg 1000, df 2

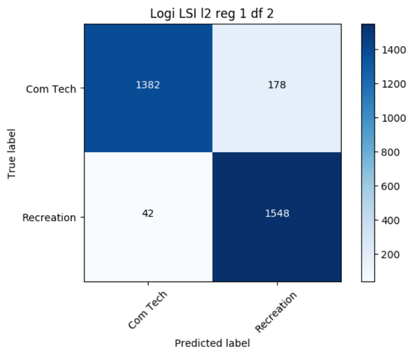
Figure 16: **Confusion Matrix of LSI with l1 norm and df 2 of Part (i)**



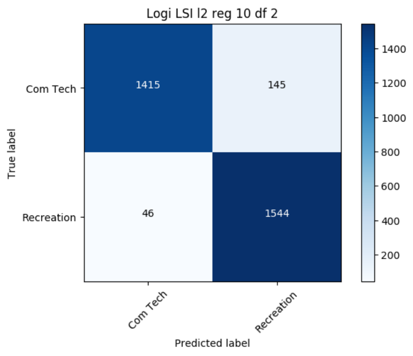
(a) LSI, l2, reg 0.01, df 2



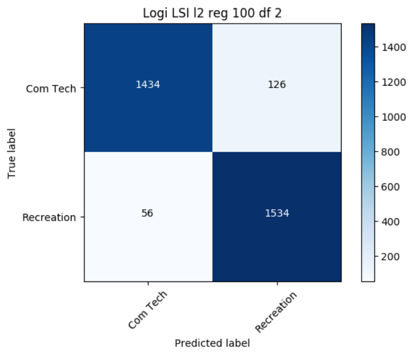
(b) LSI, l2, reg 0.1, df 2



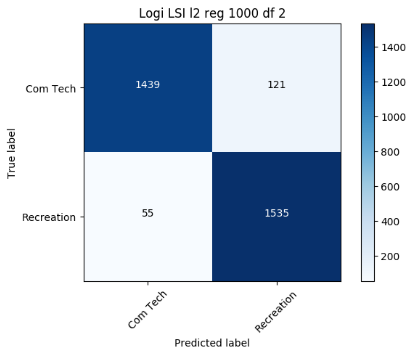
(c) LSI, l2, reg 1, df 2



(d) LSI, l2, reg 10, df 2

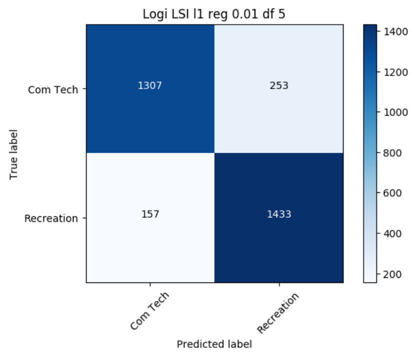


(e) LSI, l2, reg 100, df 2

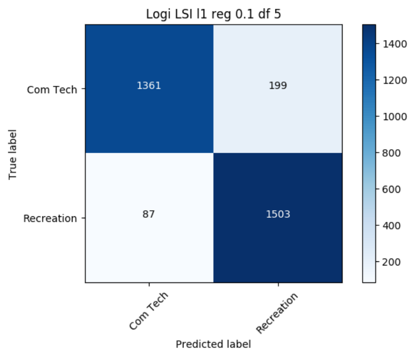


(f) LSI, l2, reg 1000, df 2

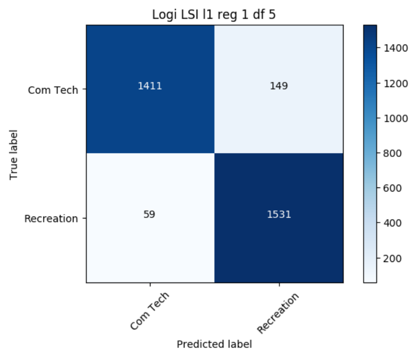
Figure 17: **Confusion Matrix of LSI with l2 norm and df 2 of Part (i)**



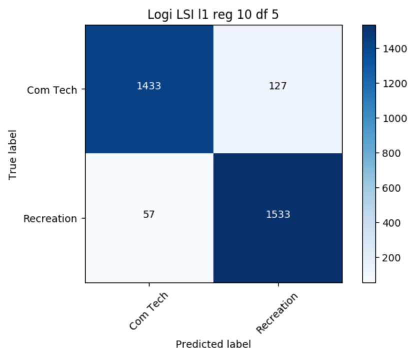
(a) LSI, l1, reg 0.01, df 5



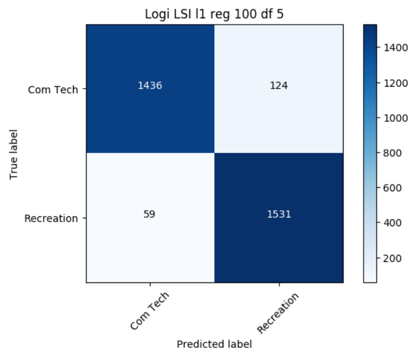
(b) LSI, l1, reg 0.1, df 5



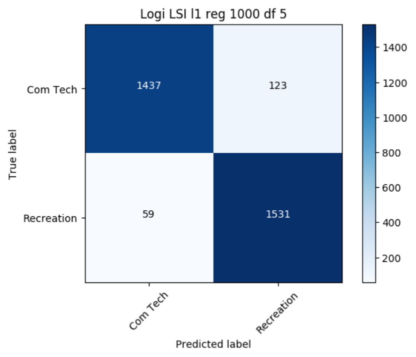
(c) LSI, l1, reg 1, df 5



(d) LSI, l1, reg 10, df 5

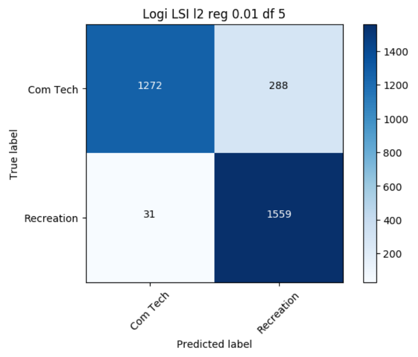


(e) LSI, l1, reg 100, df 5

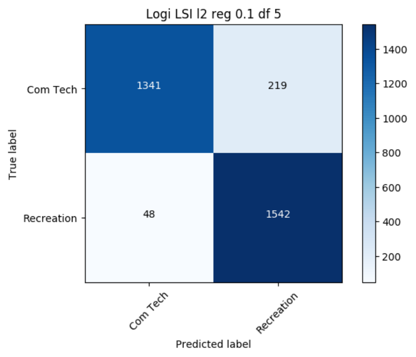


(f) LSI, l1, reg 1000, df 5

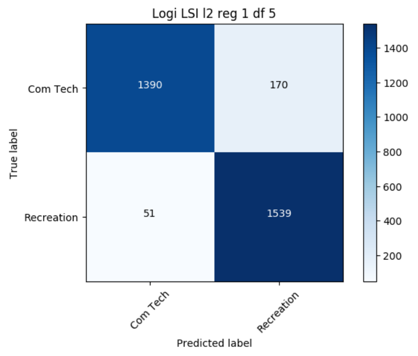
Figure 18: **Confusion Matrix of LSI with l1 norm and df 5 of Part (i)**



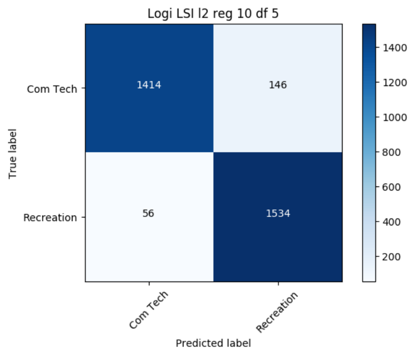
(a) LSI, l2, reg 0.01, df 5



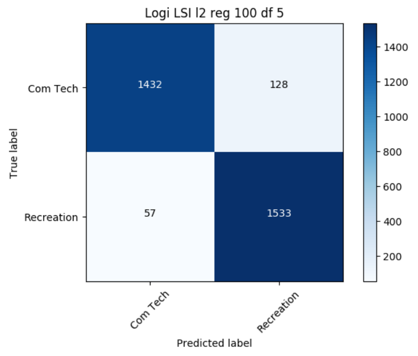
(b) LSI, l2, reg 0.1, df 5



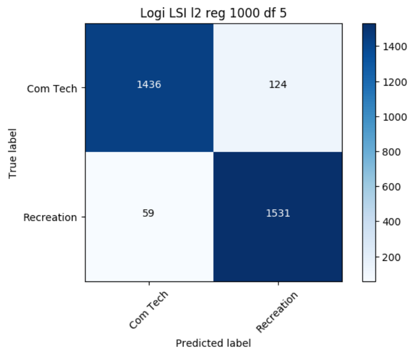
(c) LSI, l2, reg 1, df 5



(d) LSI, l2, reg 10, df 5

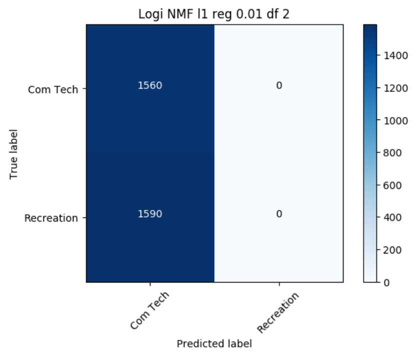


(e) LSI, l2, reg 100, df 5

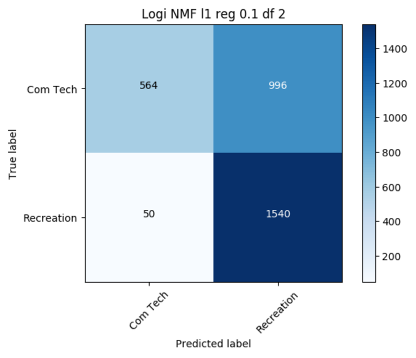


(f) LSI, l2, reg 1000, df 5

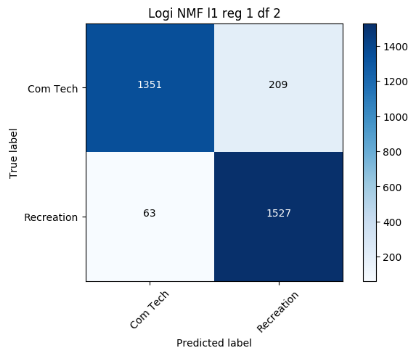
Figure 19: **Confusion Matrix of LSI with l2 norm and df 5 of Part (i)**



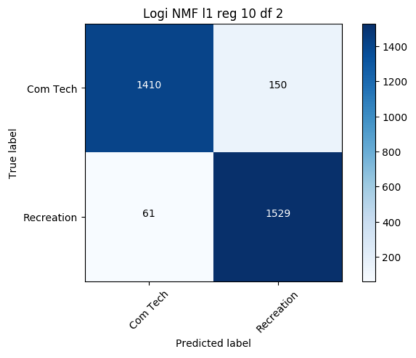
(a) NMF, l1, reg 0.01, df 2



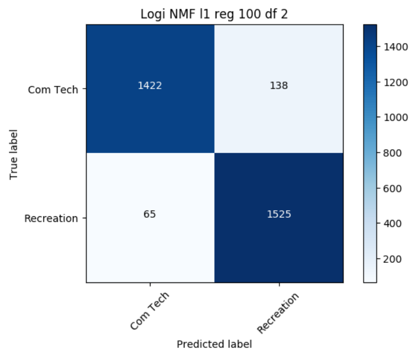
(b) NMF, l1, reg 0.1, df 2



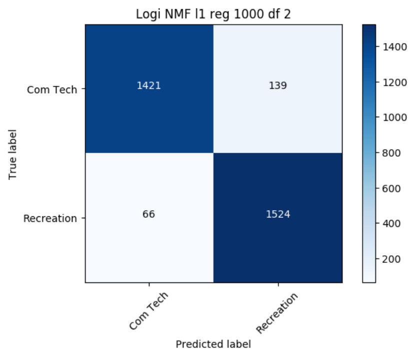
(c) NMF, l1, reg 1, df 2



(d) NMF, l1, reg 10, df 2

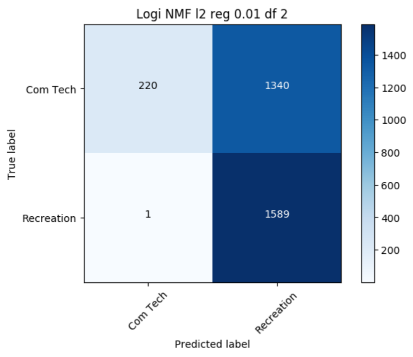


(e) NMF, l1, reg 100, df 2

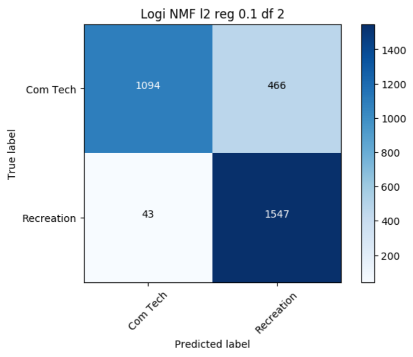


(f) NMF, l1, reg 1000, df 2

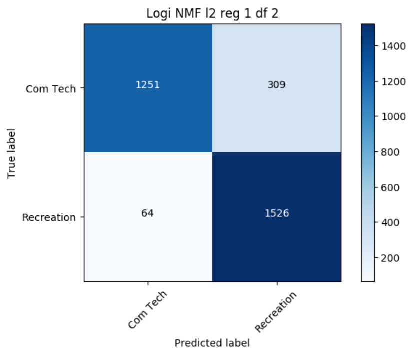
Figure 20: **Confusion Matrix of NMF with l1 norm and df 2 of Part (i)**



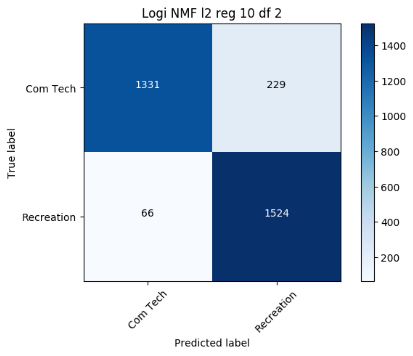
(a) LSI, l2, reg 0.01, df 2



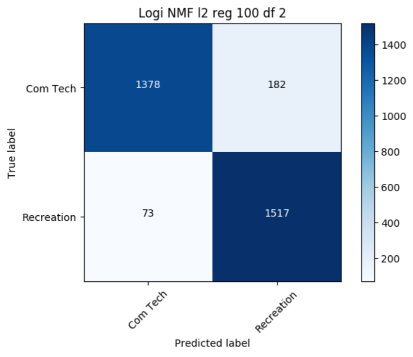
(b) LSI, l2, reg 0.1, df 2



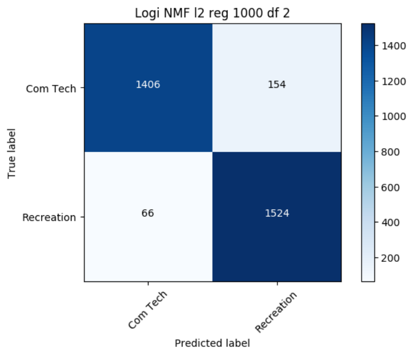
(c) LSI, l2, reg 1, df 2



(d) LSI, l2, reg 10, df 2



(e) LSI, l2, reg 100, df 2



(f) LSI, l2, reg 1000, df 2

Figure 21: **Confusion Matrix of NMF with l2 norm and df 2 of Part (i)**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.495	0.869	0.870
Recall	0.0	0.905	0.901
Preicision	0.0	0.846	0.850

Table 10: **Logistic Regression (l1, 0.01) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.683	0.904	0.910
Recall	0.953	0.943	0.945
Preicision	0.621	0.877	0.884

Table 11: **Logistic Regression (l1, 0.1) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.916	0.935	0.934
Recall	0.964	0.962	0.962
Preicision	0.880	0.914	0.911

Table 12: **Logistic Regression (l1, 1) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.932	0.940	0.941
Recall	0.964	0.965	0.964
Preicision	0.907	0.920	0.923

Table 13: **Logistic Regression (l1, 10) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.933	0.939	0.941
Recall	0.960	0.963	0.965
Preicision	0.912	0.920	0.921

Table 14: **Logistic Regression (l1, 100) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.933	0.939	0.941
Recall	0.960	0.963	0.965
Preicision	0.913	0.920	0.921

Table 15: **Logistic Regression (l1, 1000) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.575	0.896	0.900
Recall	1.0	0.983	0.981
Preicision	0.543	0.839	0.846

Table 16: **Logistic Regression (l2, 0.01) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.850	0.914	0.915
Recall	0.976	0.972	0.969
Preicision	0.781	0.872	0.875

Table 17: **Logistic Regression (l2, 0.1) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.893	0.930	0.930
Recall	0.964	0.972	0.969
Preicision	0.846	0.898	0.899

Table 18: **Logistic Regression (l2, 1) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.909	0.938	0.939
Recall	0.965	0.969	0.968
Preicision	0.869	0.914	0.917

Table 19: **Logistic Regression (l2, 10) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.924	0.940	0.939
Recall	0.966	0.967	0.962
Preicision	0.892	0.919	0.921

Table 20: **Logistic Regression (l2, 100) Metrics**

Metrics	NMF, df 2	LSI, df2	LSI, df5
Accuracy	0.933	0.939	0.940
Recall	0.964	0.963	0.965
Preicision	0.908	0.920	0.921

Table 21: **Logistic Regression (l2, 1000) Metrics**

From above 36 ROC Curve plots, 36 Confusion Matrix plots and 12 evaluation metrics tables, we can make following conclusions. First, when regularization coefficients are soft, l_2 norm regularizations works better than l_1. However, as regularization coefficients become larger and larger, l_2 loses its disavdantages and becomes similar as l_1. Second, when norm regularization is fixed, test errors decrease as regularization coefficients increase. However, when regularization coefficients are large enough, increase of coefficients does not have much effects on test errors. For example, logistic regression with l1 norm and 100 coefficients almost has no difference on performance when compared with l1 norm and 1000 coefficients.

Next, we will present tables of average of coefficients of the fitted hyperplane, and make analysis on them.

Inverse Reg Coef	NMF, df 2	LSI, df2	LSI, df5
0.01	0.0	0.1	0.1
0.1	-0.2	0.9	0.9
1	-2.6	2.2	2.3
10	-5.8	4.1	4.3
100	-10.4	4.6	4.9
1000	-12.4	4.7	4.9

Table 22: **Averaged Coefficients of HyperPlane, l1**

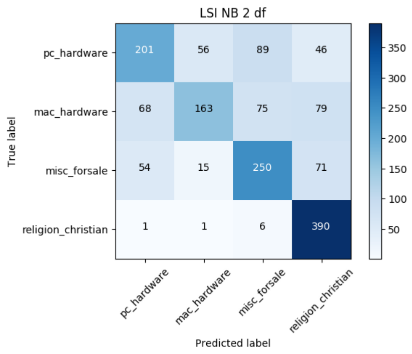
Inverse Reg Coef	NMF, df 2	LSI, df2	LSI, df5
0.01	-0.02	0.03	0.03
0.1	-0.2	0.2	0.2
1	-1.2	0.7	0.7
10	-3.5	1.7	1.8
100	-6.3	3.2	3.5
1000	-9.0	4.3	4.7

Table 23: **Averaged Coefficients of HyperPlane, l2**

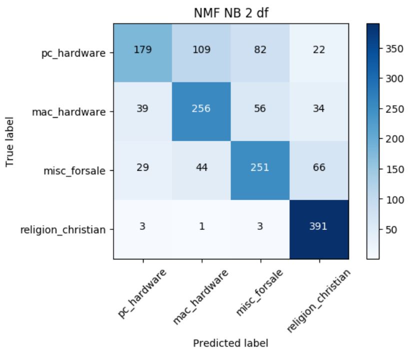
From table 22 and table 23, as the inverse of regularization coefficients increases (i.e. as regularization coefficients decreases), the magnitude of the coefficients of the fitted hyperplane increase. This is because regularization terms are in following forms: $l_1 = \lambda \sum_k |w_k|$, $l_2 = \lambda \sum_k w_i^2$. As the regularization coefficient λ decreases, coefficients of fitted hpyerplane are able to increase their magnitude and adjust the hyperplane to better position and achieve better performance.

People are interested in different type of regularization because of following reasons. l_2 regularization is computationally efficient but has non-sparse outputs. l_1 regularization is computationally inefficient but has sparse outputs. Also, different regularization coefficients have different effects on preventing overfitting. As a result, people usually choose different types of regularization based on different situations.

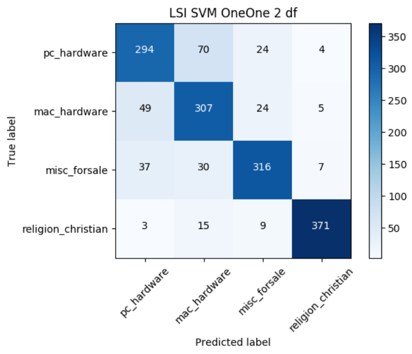
11 Part (j) Multiclass Classification



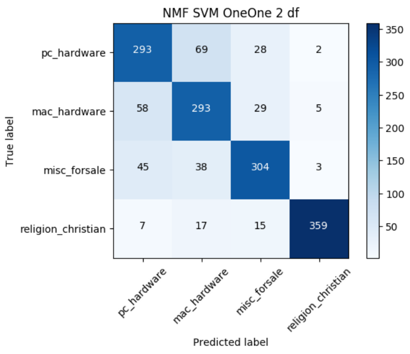
(a) LSI, Naive Bayes, df 2



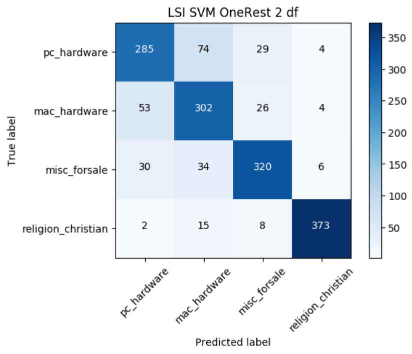
(b) NMF, Naive Bayes, df 2



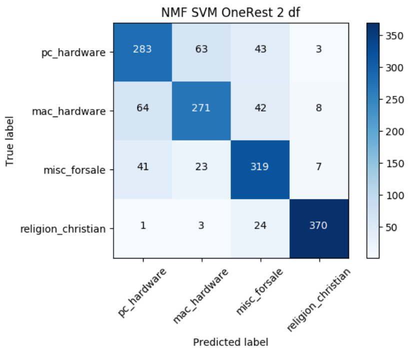
(c) LSI, OneVsOne SVM, df 2



(d) NMF, OneVsOne SVM, df 2



(e) LSI, OneVsRest SVM, df 2



(f) NMF, OneVsRest SVM, df 2

Figure 22: **Confusion Matrix of Multiclass Classification**

Metrics	Naive Bayes	OneVsOne SVM	OneVsRest SVM
Accuracy	0.642	0.823	0.818
Recall	0.642	0.823	0.818
Preicision	0.644	0.826	0.821

Table 24: **LSI Method Metrics (df 2)**

Metrics	Naive Bayes	OneVsOne SVM	OneVsRest SVM
Accuracy	0.688	0.798	0.794
Recall	0.688	0.798	0.794
Preicision	0.686	0.804	0.796

Table 25: **NMF Method Metrics (df 2)**

From confusion matrix plots and evaluation metrics tables, we can make following conclusions about multiclass classification. SVM has better performance than Naive Bayes in multiclass classification task. In particular, OneVsOne SVM has similar performance as OneVsRest performance. LSI method has better performance than NMF in Support Vector Machinne, but worse in Naive Bayes.

12 Conclusion

In this project, we research different methods for binary classification task and multiclass classification task of textual data.

In binary classification task, best Support Vector Machine classifier has similar excellent performance as best Logistic Regression classifier in both LSI and NMF dimension reduction methods. Document frequency has not much effects on performance of classifiers. As regularization coefficient decreases, fitted hyperplane is able to adjust its coefficients more flexibly and thus achieves better performance.

In multiclass classification task, SVM has better performance than Naive Bayes. In particular, OneVsOne SVM has similar performance as OneVsRest performance. LSI method has better performance than NMF in Support Vector Machinne, but worse in Naive Bayes.