ECE 219 Project 1 Report
Classification Analysis on Textual Data
Winter 2018

Hengyu Lou (005035476)
email: hylou@ucla.edu

Zhonglin Zhang (005030520)
email: evanzhang@ucla.edu

January 30, 2018

Abstract

This report mainly discusses the process of classification on textual data, which includes data loading, document modeling, feature extraction and binary classification with different methods. Meanwhile, we will use Naïve Bayes and SVM algorithms to implement multiclass classification. In addition to the realization principles and algorithms, this passage will also present the experiment results, corresponding analyses and conclusions.

1 Introduction

The main objective of this project is to learn how to use different models to realize statistical classification. For this project, we use the dataset in sklearn, "20 Newsgroup", which is a collection of approximately 20,000 newsgroup documents partitioned nearly evenly. The classifiers we use include Hard SVM, Soft SVM, Naïve Bayes, Logistic Regression, Multiclass Naïve Bayes, OneVsOne Multiclass SVM and OneVsRest MultiClass SVM.

In the following sections, we will demonstrate what we did to process the data and classify the documents correctly in the sequence of questions. Meanwhile, we will provide the obtained results and further discussions.

2 Dataset and Problem Statement

2.1 Data Loading

As mentioned above, we will use the "20 Newsgroup" dataset in sklearn. There are two methods to load this dataset from sklearn. First, we can download the dataset from URL directly to a local file using the following codes.

```
"""Script to download the 20 newsgroups text classification set"""

import os
import tarfile

try:
    from urllib import urlopen
except ImportError:
    from urllib.request import urlopen

URL = ("http://people.csail.mit.edu/jrennie/"
       "20Newsgroups/20news-bydate.tar.gz")

ARCHIVE_NAME = URL.rsplit('/', 1)[1]
TRAIN_FOLDER = "20news-bydate-train"
TEST_FOLDER = "20news-bydate-test"


if not os.path.exists(TRAIN_FOLDER) or not os.path.exists(TEST_FOLDER):

    if not os.path.exists(ARCHIVE_NAME):
        print("Downloading dataset from %s (14 MB)" % URL)
        opener = urlopen(URL)
        open(ARCHIVE_NAME, 'wb').write(opener.read())

    print("Decompressing %s" % ARCHIVE_NAME)
    tarfile.open(ARCHIVE_NAME, "r:gz").extractall(path='.')
    os.remove(ARCHIVE_NAME)
```

while it is easier to use the built-in dataset loader function '`fetch_20newsgroups`'. Because we will mainly use only 8 subclasses out of two major classes 'Computer Technology' and 'Recreational Activity' as listed in the Table 1, our group uses the following codes to complete the data loading.

Table 1: 8 Subclasses of 2 Major Classes

Computer technology	**Recreational activity**
comp.graphics	rec.autos
comp.os.ms-windows.misc	rec.motorcycles
comp.sys.ibm.pc.hardwre	rec.sport.baseball
comp.sys.mac.hardware	rec.sport.hockey

In [1]:
```
from sklearn.datasets import fetch_20newsgroups
categories = ['comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', \
              'comp.sys.mac.hardware', 'rec.autos', 'rec.motorcycles', \
              'rec.sport.baseball', 'rec.sport.hockey']
# categories = ['comp.graphics', 'comp.sys.mac.hardware']
twenty_train = fetch_20newsgroups(subset='train', categories=categories, shuffle=True, \
                                  random_state=42, remove=('headers'))
twenty_test = fetch_20newsgroups(subset='test', categories=categories, shuffle=True, \
                                 random_state=42, remove=('headers'))
print 'Fetching data done!'
```

```
Fetching data done!
```

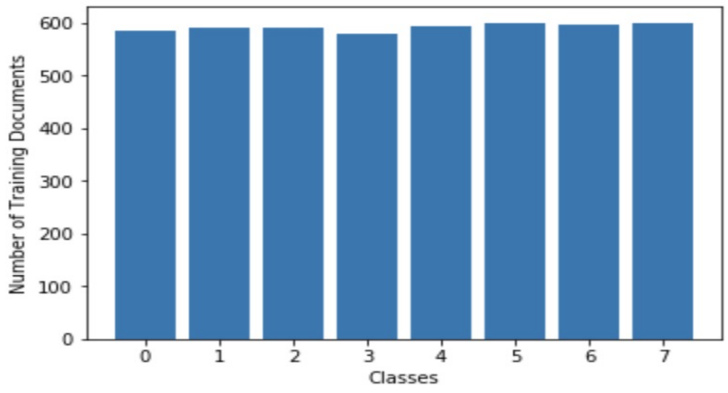
2.2 Question a: Histogram Plotting

In order to guarantee the proper classification, we should make sure first that the training documents are distributed evenly in different classes. Otherwise, we have to handle the imbalance problem by modifying the penalty function or down-sampling the majority classes. Hence, first of all, we plot the histogram of the number of the training data and check if they are distributed evenly. The histogram is shown in Figure 1. According to the shown result, we can conclude that the data set has been balanced already.

3 Modeling Text Data and Feature Extraction

3.1 Question b: TFxIDF Vector Representations

In this step, we need to preprocess the textual data, which includes:




```
Part a) done
```

Figure 1: Histogram of the number of the training documents

1. Tokenize each document into words
2. Exclude the stop words, punctuations
3. Convert all words into stemmed version
4. Represent the documents with TFxIDF vectors

Here, we use Lemmatizer, a WorldNet's built-in morphy function, to map the different forms of the same word to the same root word.
Secondly, TFxIDF vector representation is defined as:

$$TFxIDF(t,d) = tf(t,d) * idf(t)$$

where $tf(t,d)$ represents the frequency of term t in document d, and inverse document frequency is defined as:

$$idf(t) = log[n/df(t)] + 1$$

where n is the total number of documents and $d(f)$ is the number of documents that contain the term t.
For this project, we set **min_df=2** and **min_df=5** respectively to complete the following questions. Hence, for question b, the numbers of extracted term in these two conditions are respectively:

$$Num_{min\_df=2} = 23313$$
$$Num_{min\_df=5} = 9591$$

3.2 Question c: TFxICF Vector Representation

In this question, we want to quantify the significance of each word to a class. In order to solve this problem, we use a measure named TFxICF whose definition is:

$$TFxICF(t,c) = tf(t,c) * icf(t)$$

where $tf(t,c)$ represents the term frequency in a class c, and inverse class frequency $icf(t)$ is defined as:

$$icf(t) = log[n_{classes}/cf(t)] + 1$$

Similar to the definitions in TFxIDF, $n_{classes}$ is the total number of classes, and $cf(t)$ is the class frequency, the number of classes within which there is at least a document containing the term t. By calculating the TFxICF representation of each term, we can obtain the 10 most significant terms in the given 4 classes. The calculation results are listed in the Table 2. Note that in this part, we use the unbalance dataset of 20 classes.

Table 2: 10 Most Significant Terms in 4 Classes

comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	misc.forsale	soc.religion.christian
scsi	edu	edu	god
edu	line	line	edu
drive	mac	subject	christian
line	subject	sale	say
com	organization	organization	church
ide	use	post	subject
subject	apple	university	jesus
use	quadra	com	people
organization	problem	new	line
card	post	host	christ

4 Feature Selection

4.1 Question d: LSI and NMF

After the above operations, the dimensionality of the TFxIDF vectors generally ranges in the order of thousands while the matrix is actually sparse and low-rank. With the aim of improving learning algorithm performance, we need to transform the features into a lower dimensional space by some mathematical methods. Here, we use two methods to reduce the matrix dimension separately: Latent Semantic Indexing (LSI) and Non-Negative Matrix Factorization (NMF). We can implement the decomposition using the following codes

```
# LSI
from sklearn.decomposition import TruncatedSVD
svd = TruncatedSVD(n_components=50, random_state=42)
X_train_lsi = svd.fit_transform(X_train_tfidf)
X_test_lsi = svd.fit_transform(X_test_tfidf)

# NMF
from sklearn.decomposition import NMF
nmf = NMF(n_components=50, init='random', random_state=42)
X_train_nmf = nmf.fit_transform(X_train_tfidf)
X_test_nmf = nmf.fit_transform(X_test_tfidf)
```

By the above methods, we can obtain a 50-dimensional vector for each document and continue the classification with them.

5 Learning Algorithm

5.1 Question e: Hard SVC and Soft SVC

In this part, we need to use both hard margin and soft margin SVM classifier (SVC) to separate the documents into two groups. For hard SVC, we should set γ to 1000 and for soft SVC, 0.001. With different decomposition methods and mid_df values, there are 6 combinations in total. So we will list these experiment results in the following subsections. Note that for the SVC, we set the kernel to linear type.

5.1.1 Hard SVC

1. LSI with min_df = 2

```
Accuracy of LSI min_df = 2 is 0.971111111111
Precision of LSI min_df = 2 is 0.966397013068
Recall of LSI min_df = 2 is 0.976729559748
```

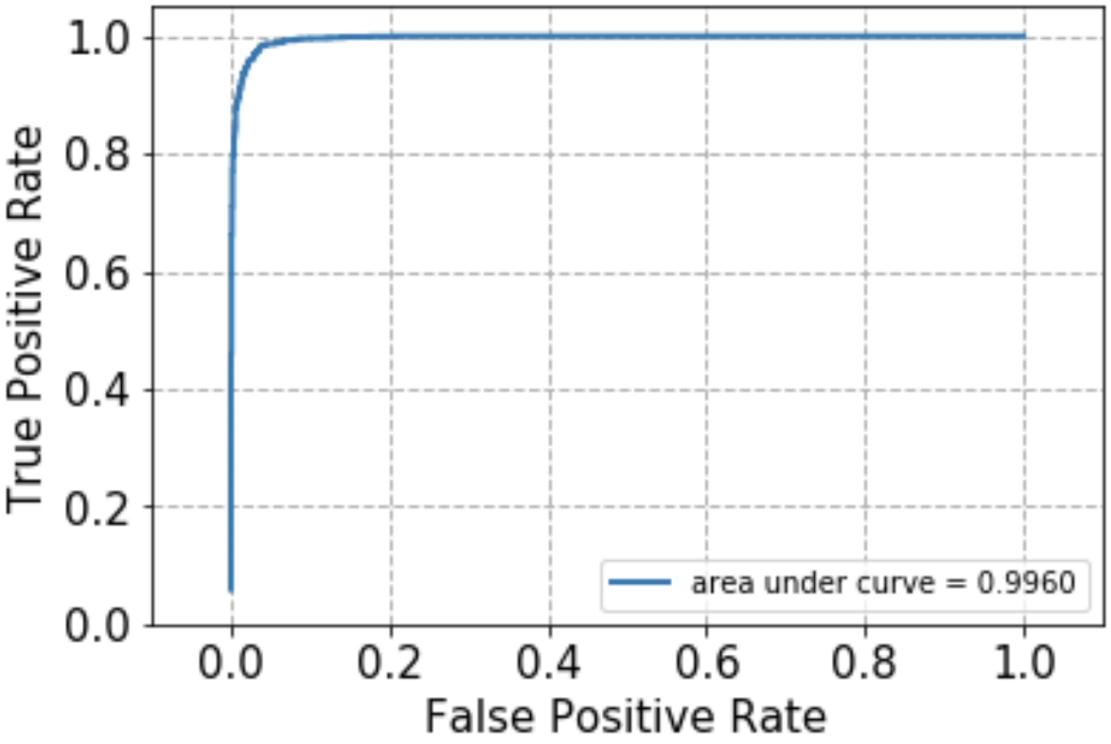


Figure 2: ROC Curve($\gamma = 1000$, min_df=2, LSI)

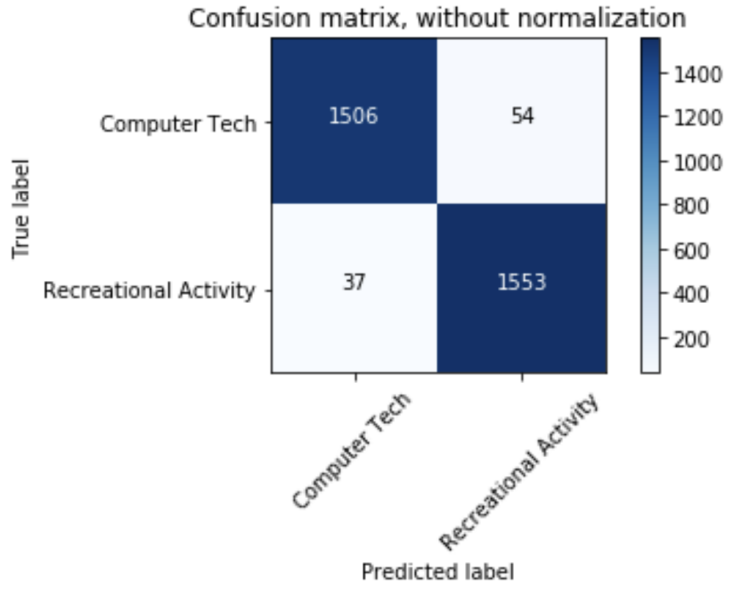


Figure 3: Unnormalized Confusion Matrix($\gamma = 1000$, min_df=2, LSI)

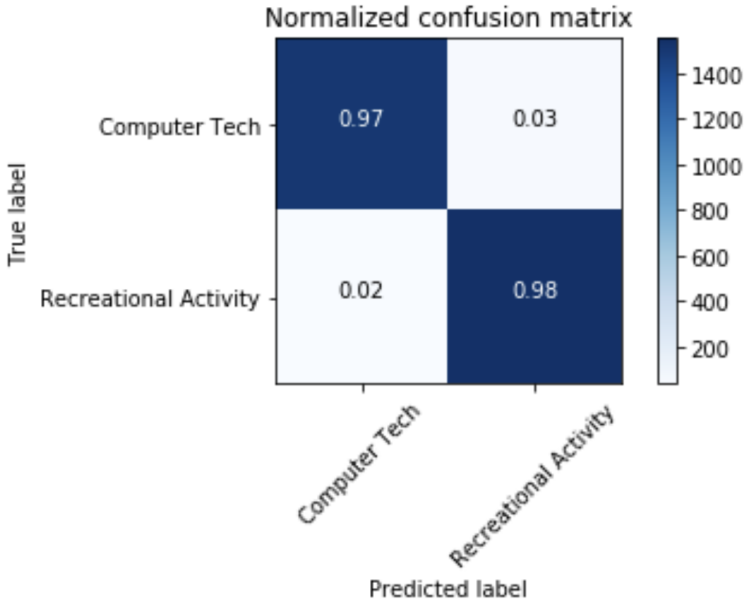


Figure 4: Normalized Confusion Matrix($\gamma = 1000$, min_df=2, LSI)

2. LSI with min_df = 5

```
Accuracy of LSI min_df = 5 is 0.971428571429
Precision of LSI min_df = 5 is 0.96468401487
Recall of LSI min_df = 5 is 0.979245283019
```

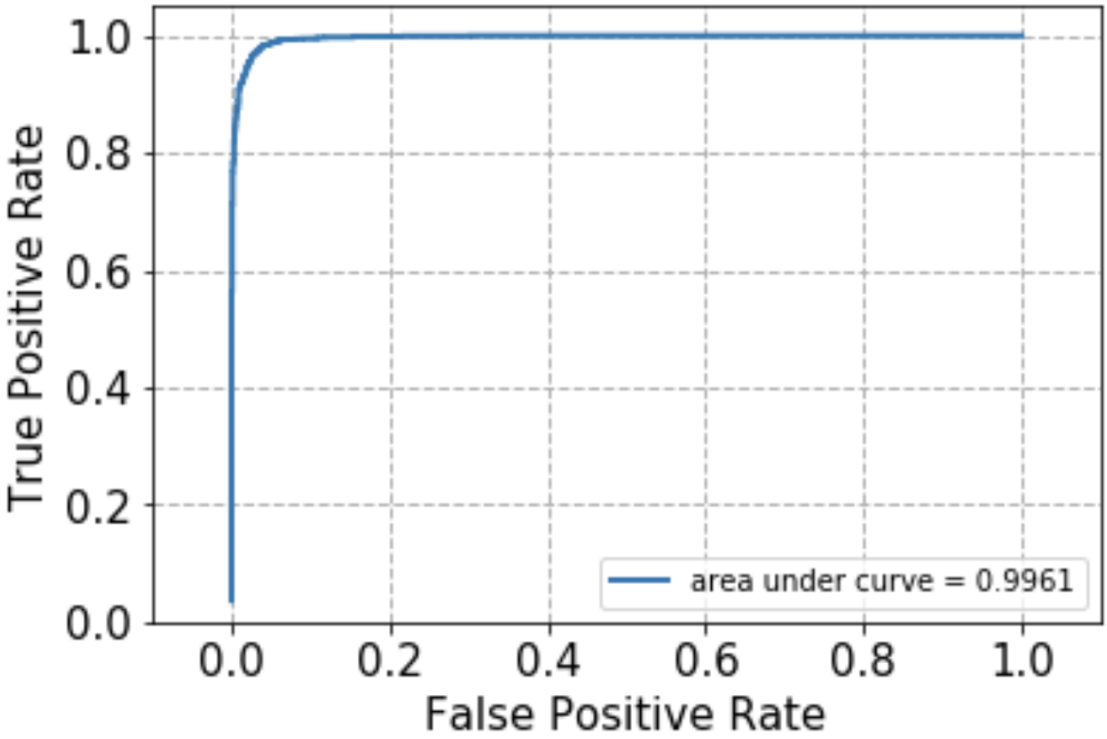


Figure 5: ROC Curve($\gamma = 1000$, min_df=5, LSI)

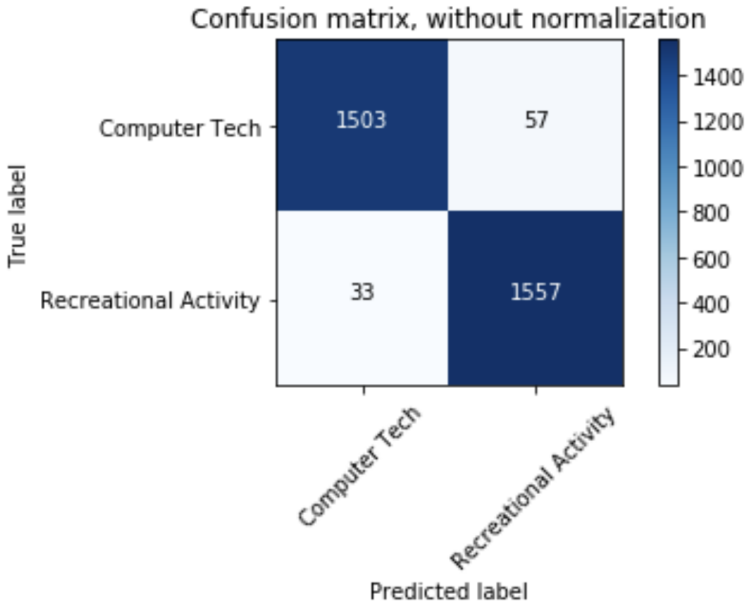


Figure 6: Unnormalized Confusion Matrix($\gamma = 1000$, min_df=5, LSI)

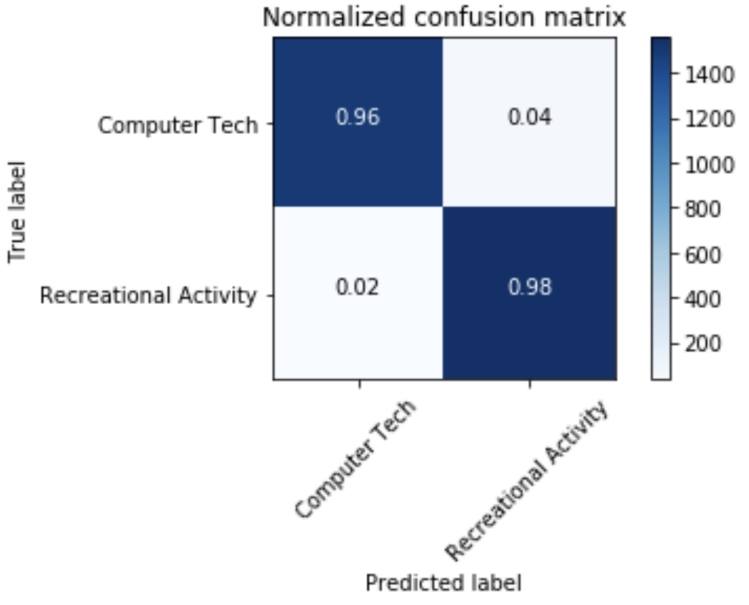


Figure 7: Normalized Confusion Matrix($\gamma = 1000$, min_df=5, LSI)

3. NMF with min_df = 2

```
Accuracy of NMF min_df = 2 is 0.969841269841
Precision of NMF min_df = 2 is 0.962275819419
Recall of NMF min_df = 2 is 0.978616352201
```

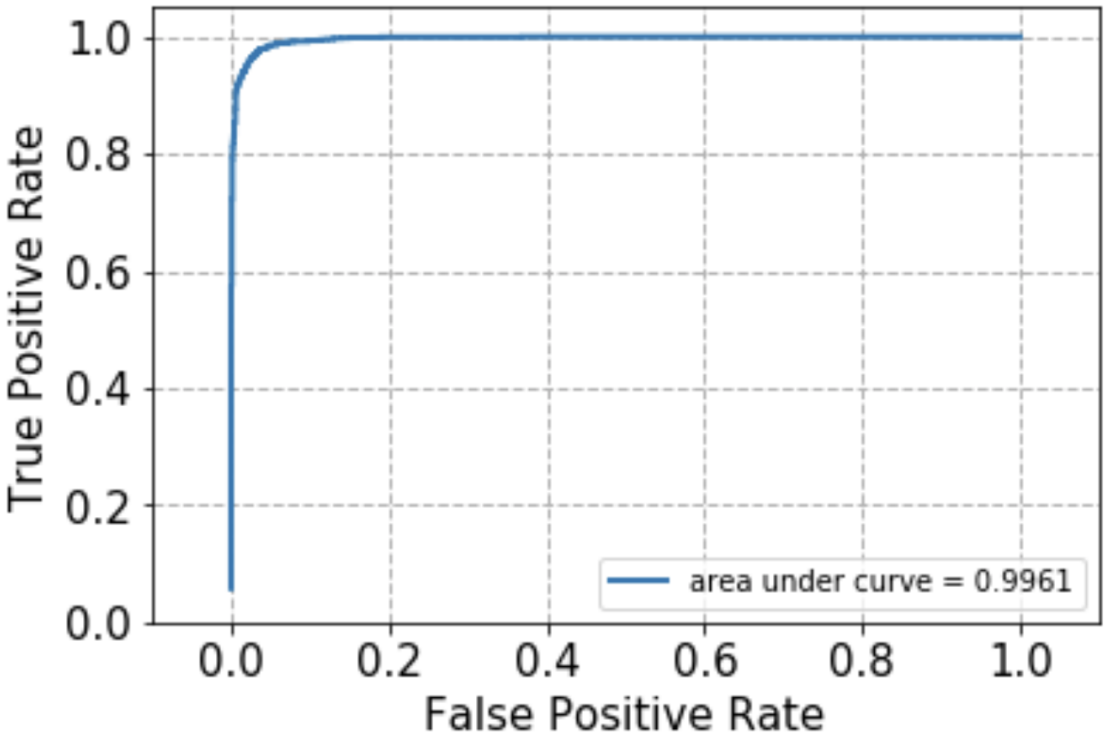


Figure 8: ROC Curve($\gamma = 1000$, min_df=2, NMF)

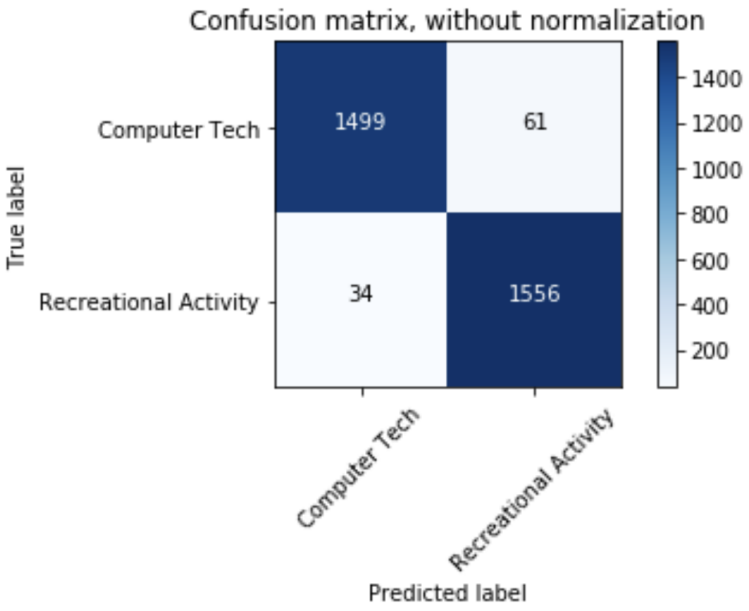


Figure 9: Unnormalized Confusion Matrix($\gamma = 1000$, min_df=2, NMF)

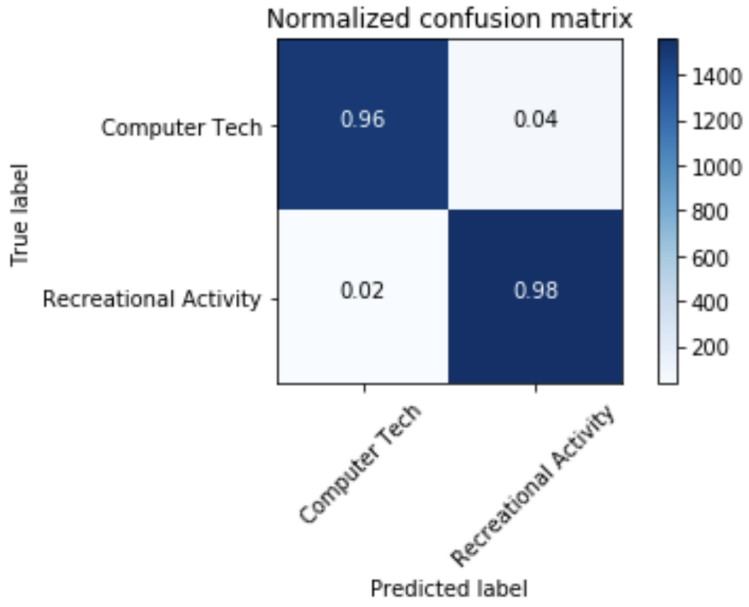


Figure 10: Normalized Confusion Matrix($\gamma = 1000$, min_df=2, NMF)

5.1.2 Soft SVC

1. LSI with min_df = 2

```
Accuracy of LSI min_df = 2 is 0.971428571429
Precision of LSI min_df = 2 is 0.96468401487
Recall of LSI min_df = 2 is 0.979245283019
```

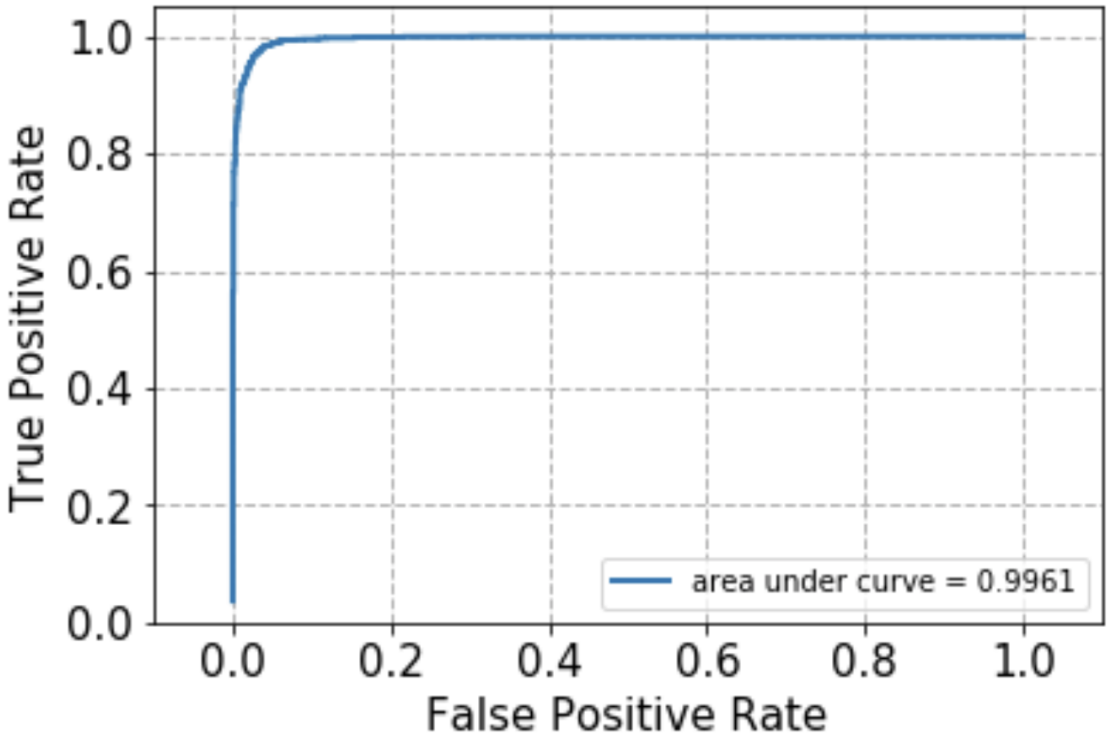


Figure 11: ROC Curve($\gamma = 0.001$, min_df=2, LSI)

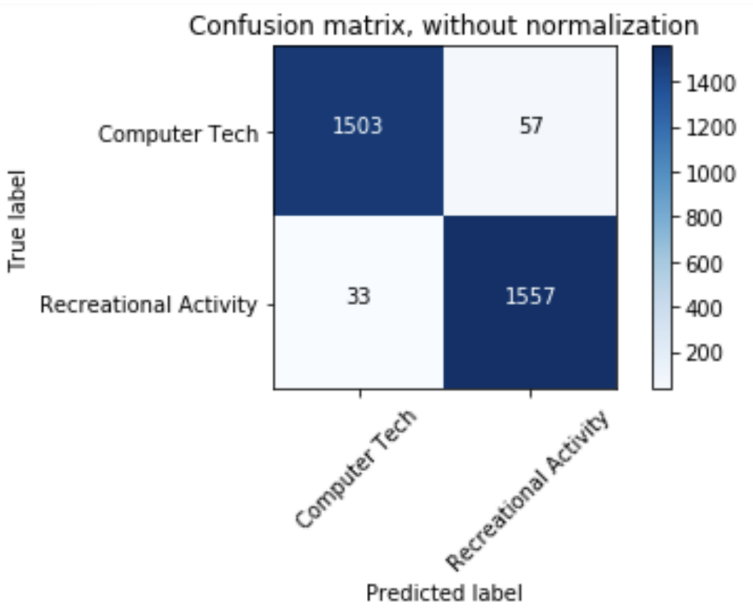


Figure 12: Unnormalized Confusion Matrix($\gamma = 0.001$, min_df=2, LSI)

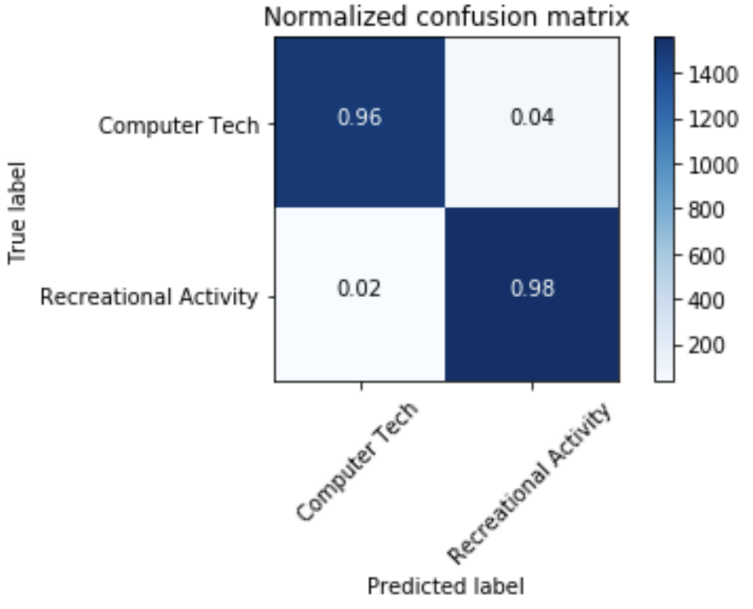


Figure 13: Normalized Confusion Matrix($\gamma = 0.001$, min_df=2, LSI)

2. LSI with min_df = 5

```
Accuracy of LSI min_df = 5 is 0.971428571429
Precision of LSI min_df = 5 is 0.96468401487
Recall of LSI min_df = 5 is 0.979245283019
```

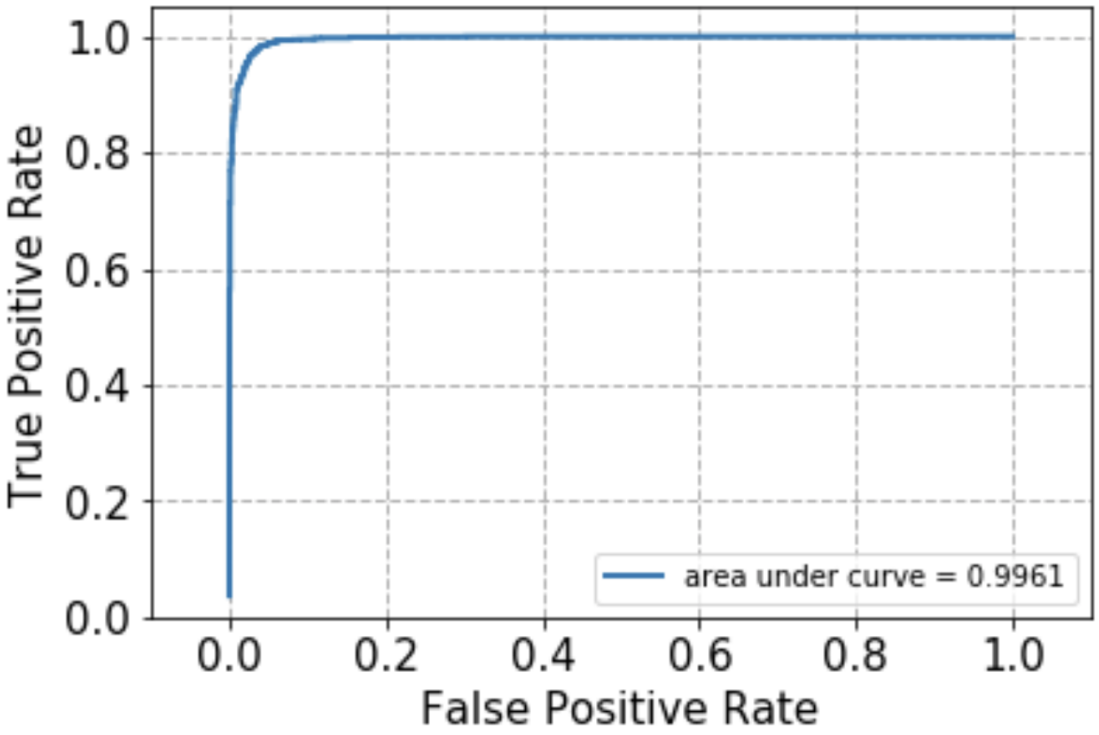


Figure 14: ROC Curve($\gamma = 0.001$, min_df=5, LSI)

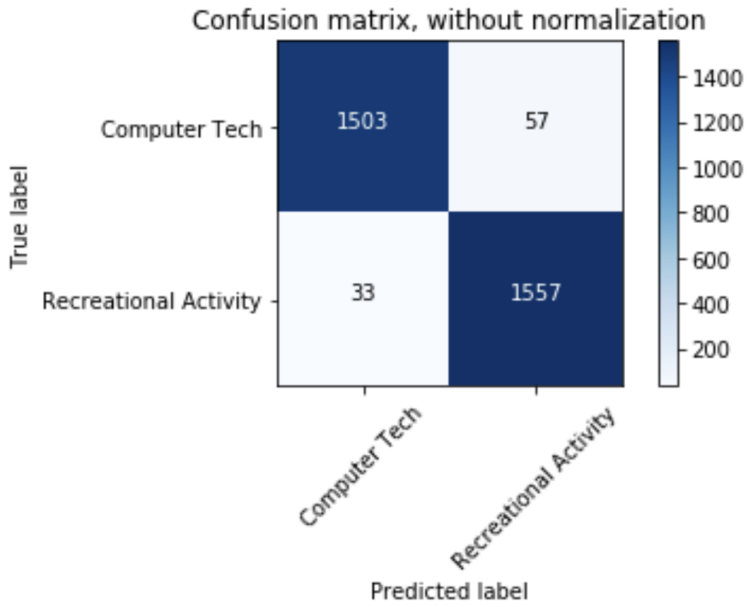


Figure 15: Unnormalized Confusion Matrix($\gamma = 0.001$, min_df=5, LSI)

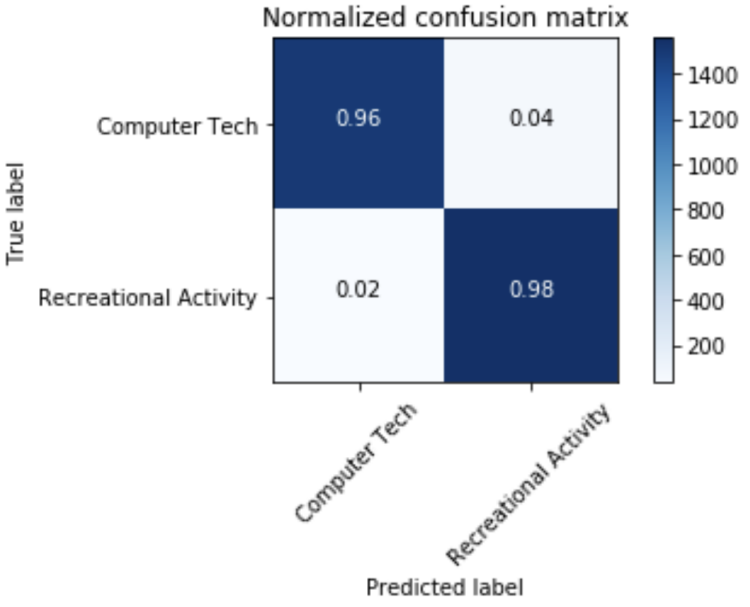


Figure 16: Normalized Confusion Matrix($\gamma = 0.001$, min_df=5, LSI)

3. NMF with min_df = 2

```
Accuracy of NMF min_df = 2 is 0.969841269841
Precision of NMF min_df = 2 is 0.962275819419
Recall of NMF min_df = 2 is 0.978616352201
```

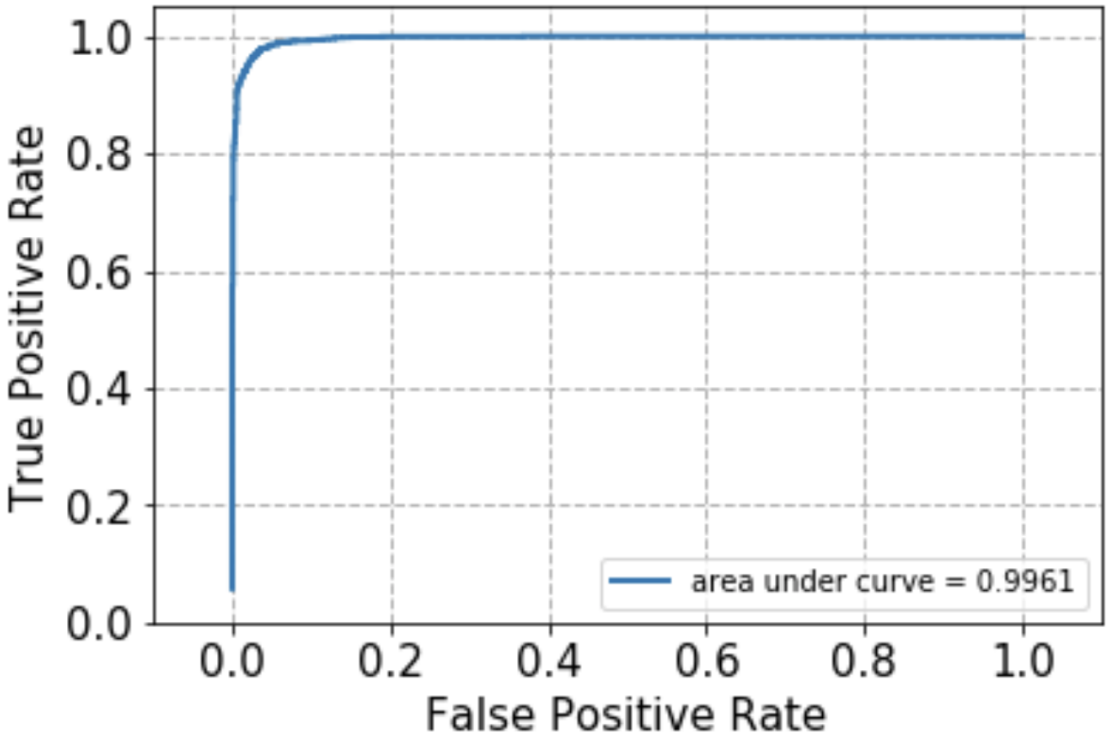


Figure 17: ROC Curve($\gamma = 0.001$, min_df=2, NMF)

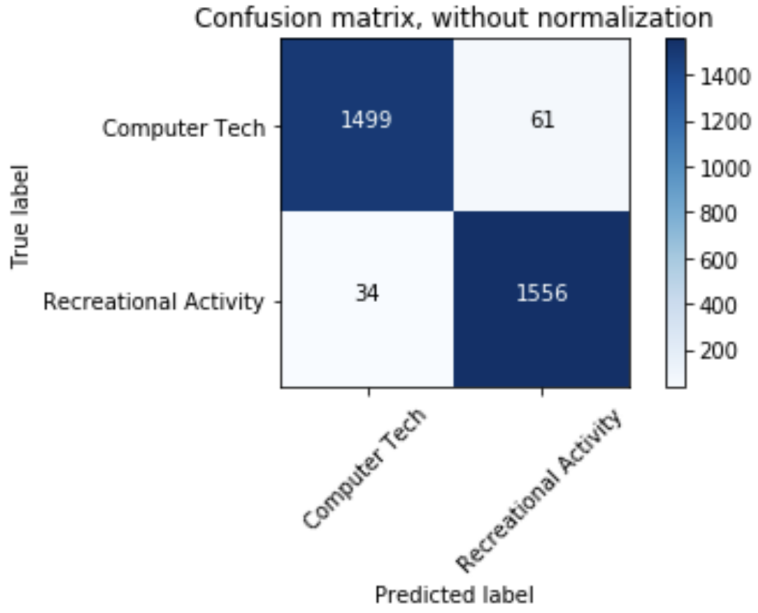


Figure 18: Unnormalized Confusion Matrix($\gamma = 0.001$, min_df=2, NMF)

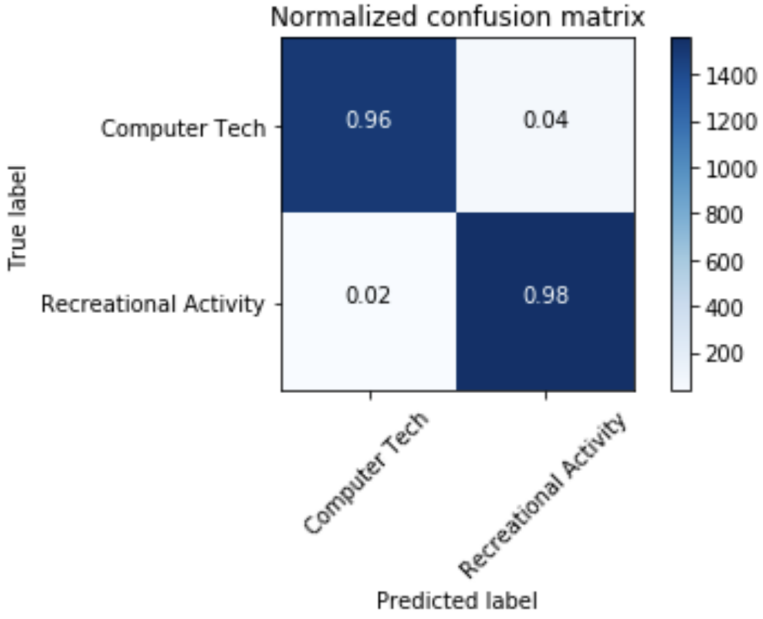


Figure 19: Normalized Confusion Matrix($\gamma = 0.001$, min_df=2, NMF)

5.2 Question f: SVC with 5-Fold Cross-Validation

For this part, we will use 5-fold cross-validation to determine γ value to achieve the best classification performance. We swept γ from 10^{-3} to 10^{3}. According to our results, the best value of γ of condtion "LSI min_df = 2" is 100, for condtion "LSI min_df = 2" is 100 and for condition "NMF min_df = 2" is also 100. Then repeat the processes in Question e and the results are listed below.

1. LSI with min_df = 2

```
Accuracy of LSI min_df = 2 is 0.969206349206
Precision of LSI min_df = 2 is 0.958256599141
Recall of LSI min_df = 2 is 0.981761006289
```

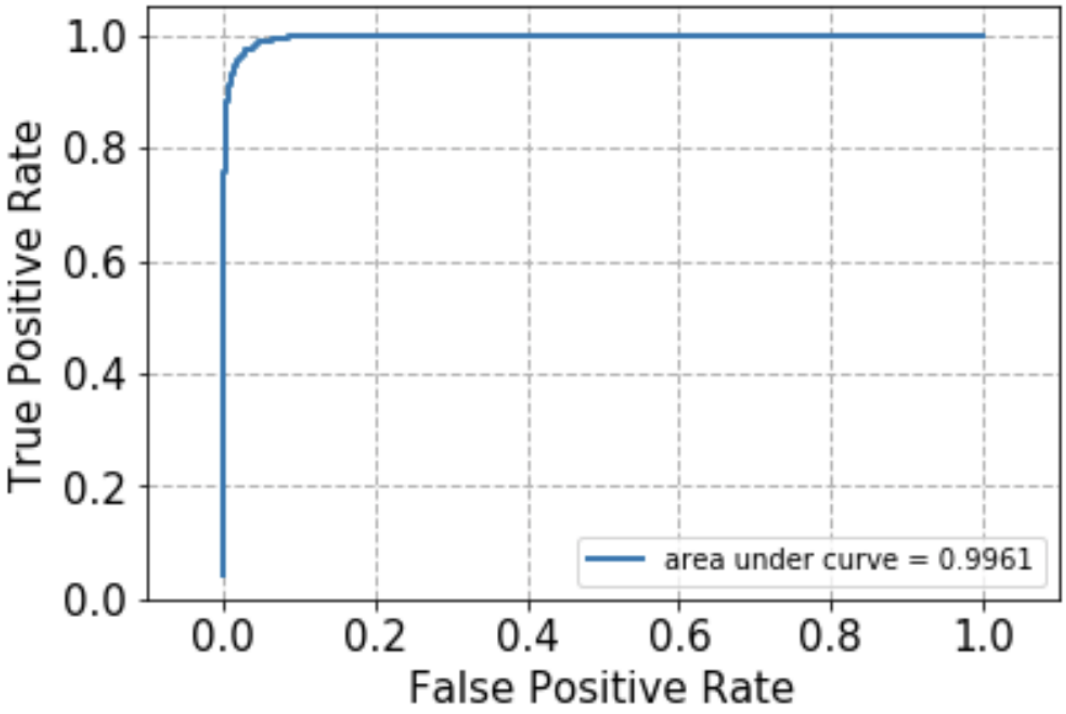


Figure 20: ROC Curve($\gamma = 100$, min_df=2, LSI)

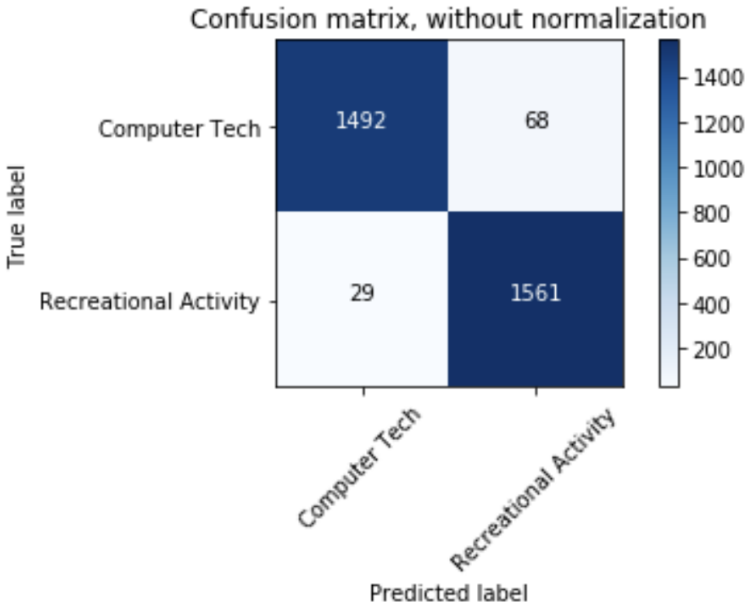


Figure 21: Unnormalized Confusion Matrix($\gamma = 100$, min_df=2, LSI)

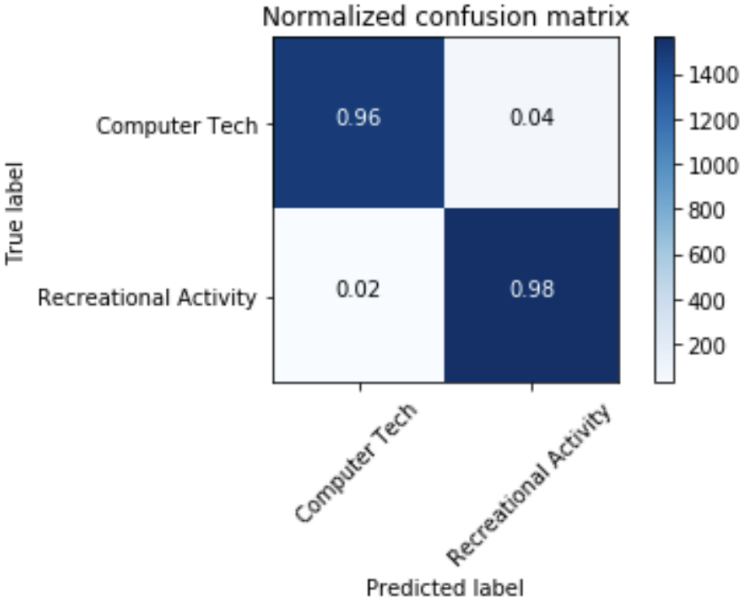


Figure 22: Normalized Confusion Matrix($\gamma = 100$, min_df=2, LSI)

2. LSI with min_df = 5

```
Accuracy of LSI min_df = 5 is 0.969206349206
Precision of LSI min_df = 5 is 0.958819913952
Recall of LSI min_df = 5 is 0.981132075472
```

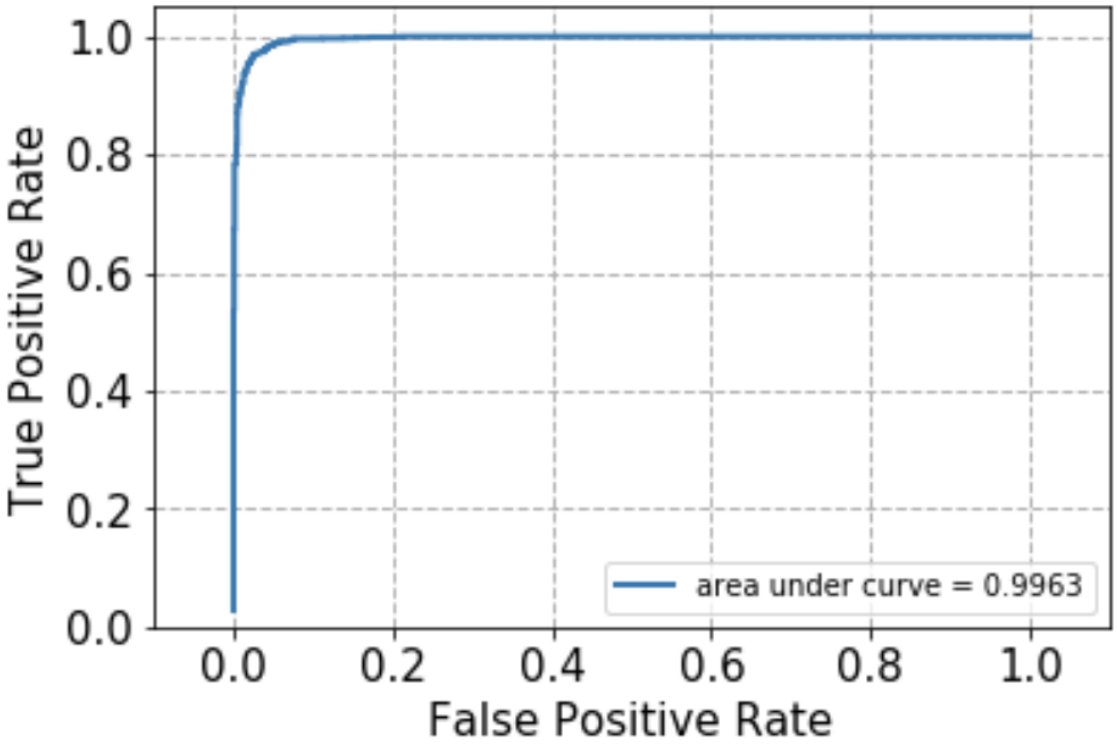


Figure 23: ROC Curve($\gamma = 100$, min_df=5, LSI)

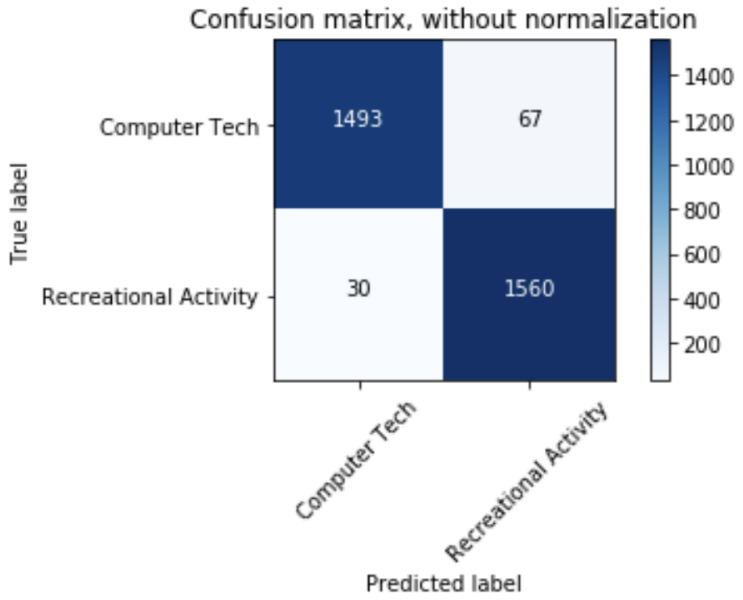


Figure 24: Unnormalized Confusion Matrix($\gamma = 100$, min_df=5, LSI)

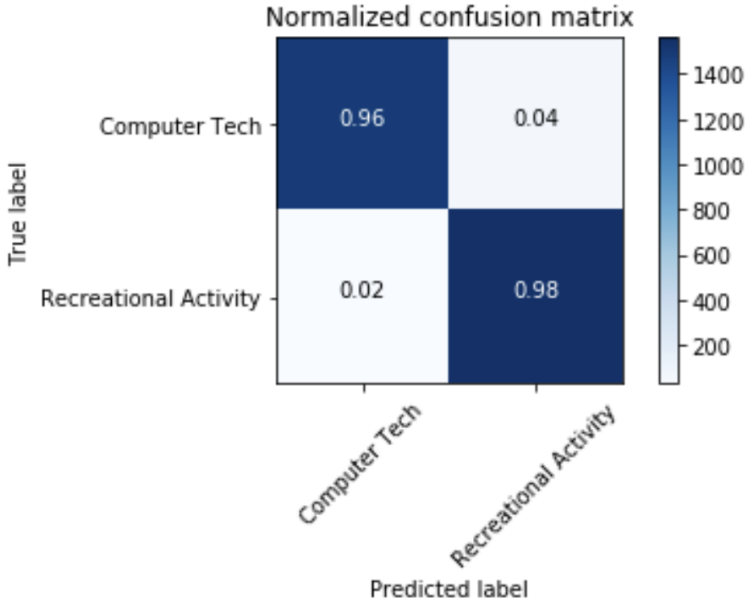


Figure 25: Normalized Confusion Matrix($\gamma = 100$, min_df=5, LSI)

3. NMF with min_df = 2

```
Accuracy of NMF min_df = 2 is 0.954920634921
Precision of NMF min_df = 2 is 0.935096153846
Recall of NMF min_df = 2 is 0.978616352201
```

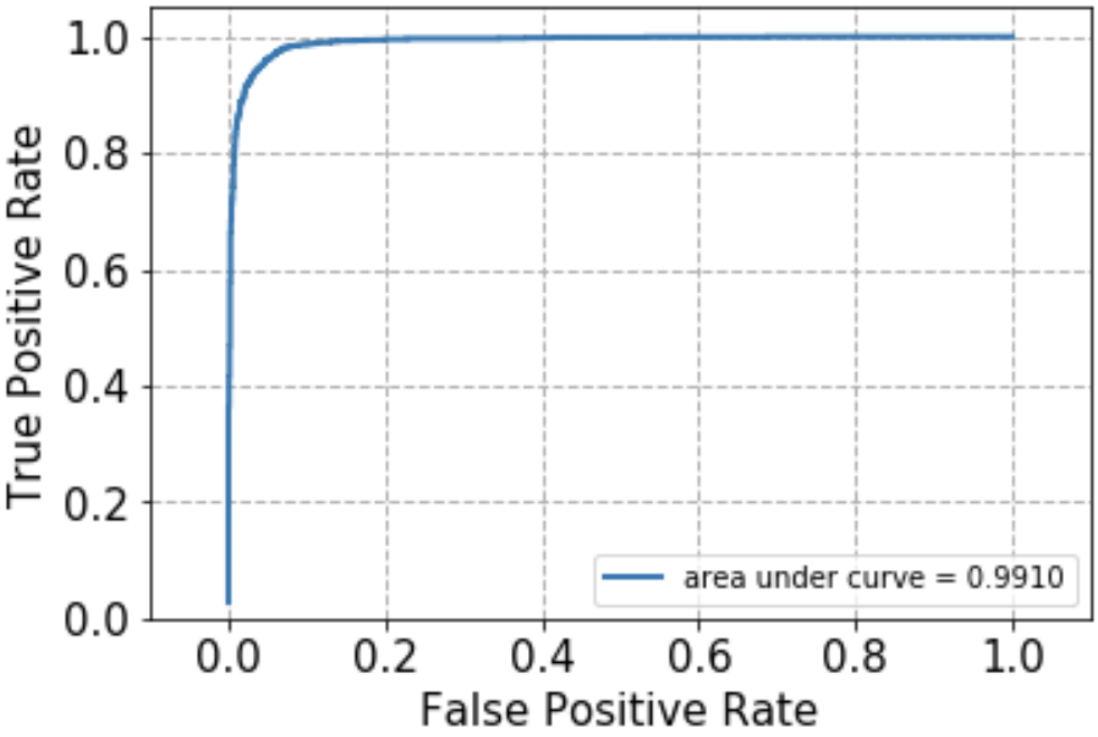


Figure 26: ROC Curve($\gamma = 100$, min_df=2, NMF)

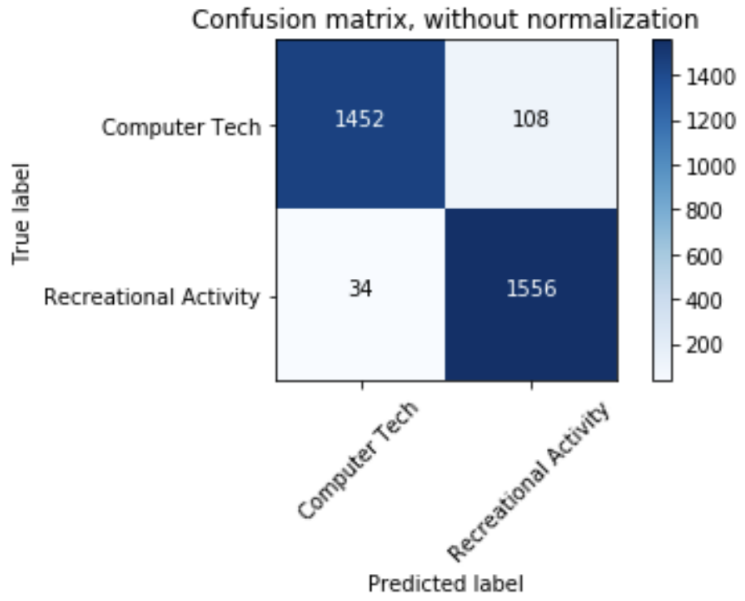


Figure 27: Unnormalized Confusion Matrix($\gamma = 100$, min_df=2, NMF)

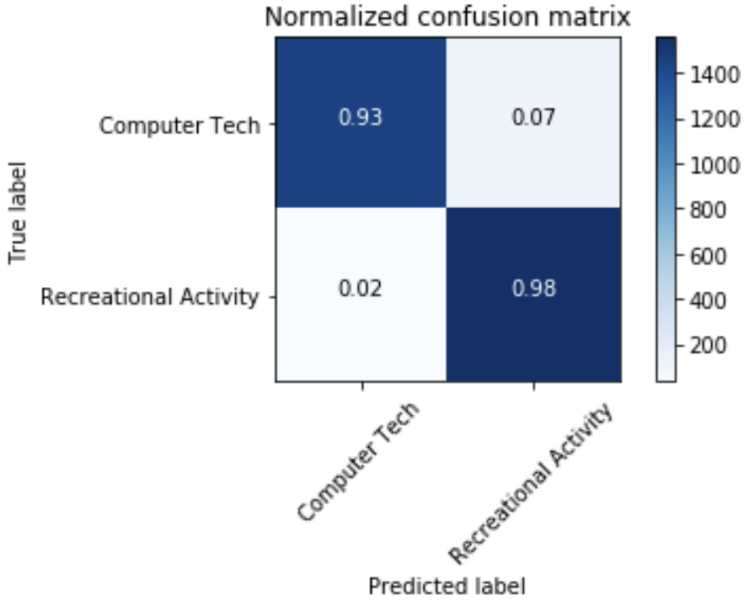


Figure 28: Normalized Confusion Matrix($\gamma = 100$, min_df=2, NMF)

5.3 Question g: Naïve Bayes Classifier

The third kind of classifier we use is Naïve Bayes algorithm. This algorithm estimates the maximum likelihood probability of a class given a document using Bayes rule. The ROC curve and related parameters are shown as follow. We should note that LSI method is not applicable for this this part.

```
Accuracy of NMF min_df = 2 is 0.944126984127
Precision of NMF min_df = 2 is 0.90632183908
Recall of NMF min_df = 2 is 0.991823899371
```

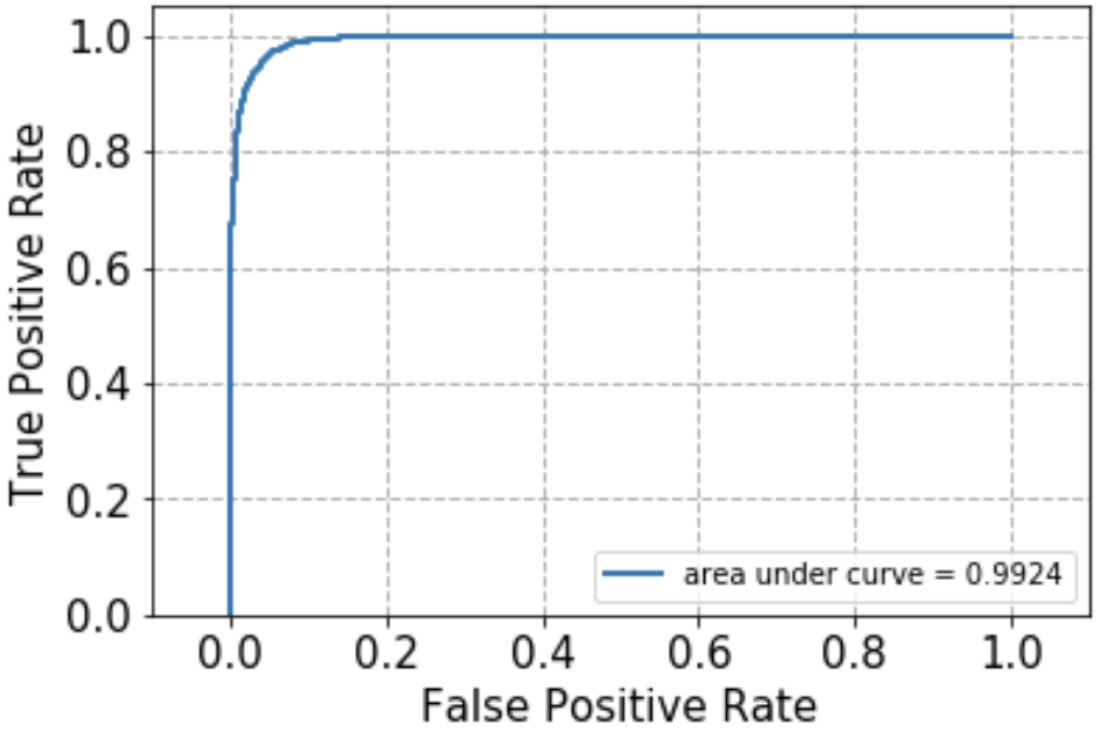


Figure 29: ROC Curve(min_df=2, NMF)

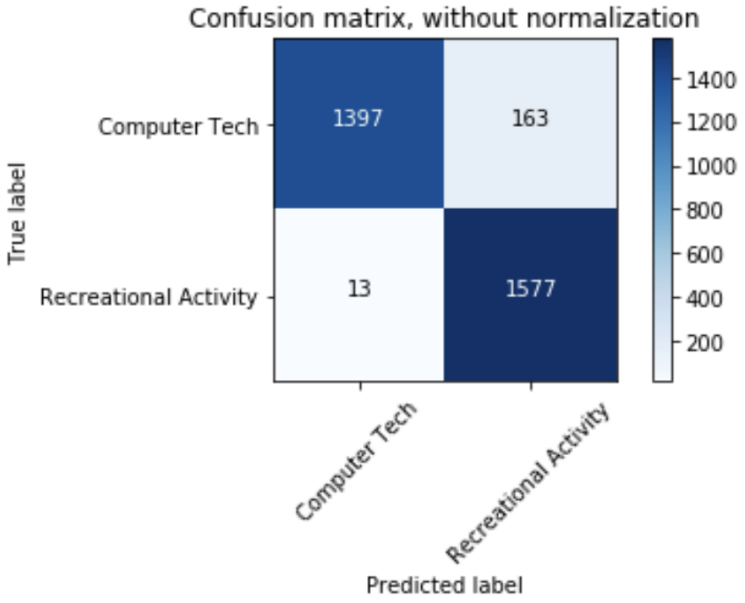


Figure 30: Unnormalized Confusion Matrix(min_df=2, NMF)

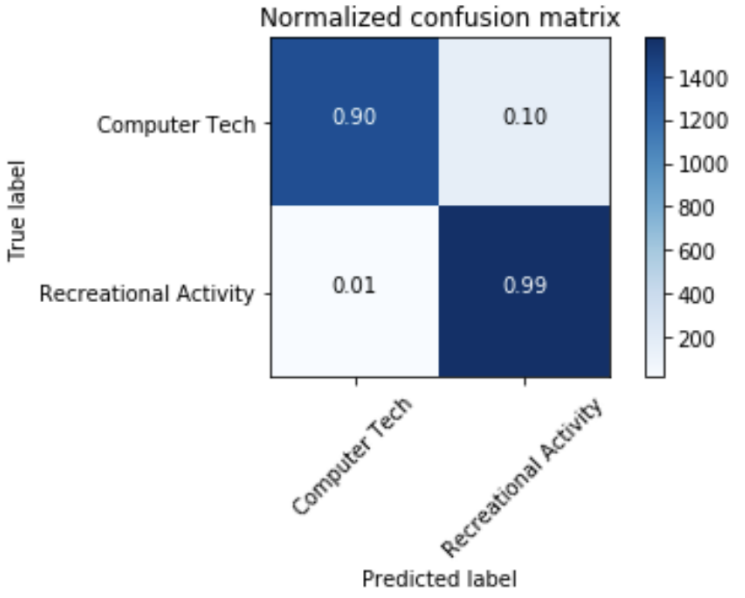


Figure 31: Normalized Confusion Matrix(min_df=2, NMF)

5.4 Question h: Logistic Regression Classifier

The fourth kind of classifier we use is logistic regression classifier. We repeat plotting the ROC curve and calculating related parameters as before. Note that in order to implement the logistic regression classifier without regularization term, we set the parameter C to an extreme large positive number (10,000,000) with the statement: `'clf', LogisticRegression(C=10000000)` to simulate the required condition. When C is big enough, the value of γ, which is the reciprocal of C, is fairly small so that the effect of regularization is eliminated basically.

1. LSI with min_df = 2
   ```
   Accuracy of LSI is 0.972698412698
   Precision of LSI is 0.963054187192
   Recall of LSI is 0.983647798742
   ```

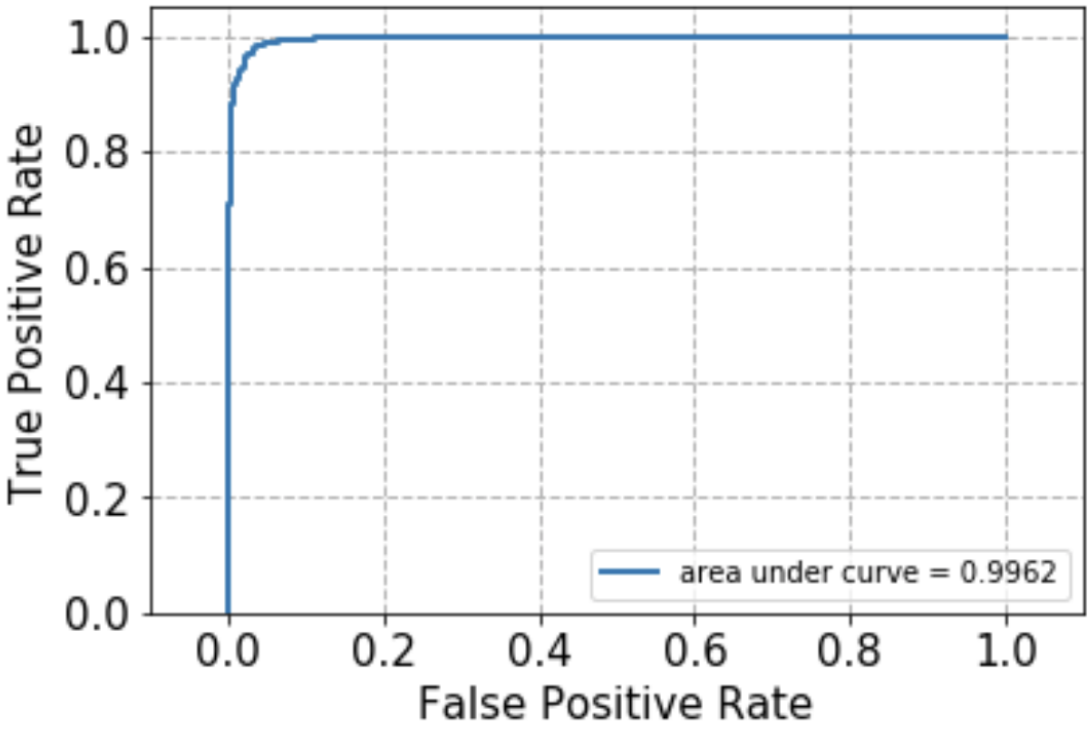


Figure 32: ROC Curve(min_df=2, LSI)

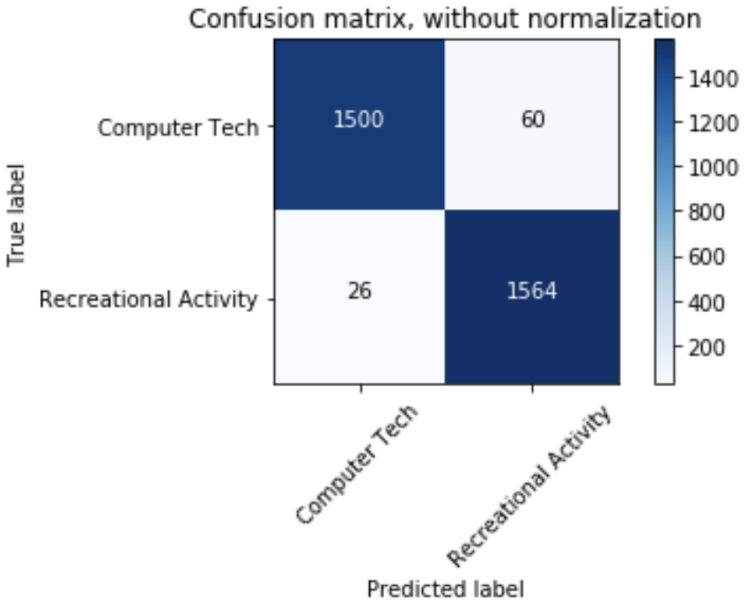


Figure 33: Unnormalized Confusion Matrix(min_df=2, LSI)

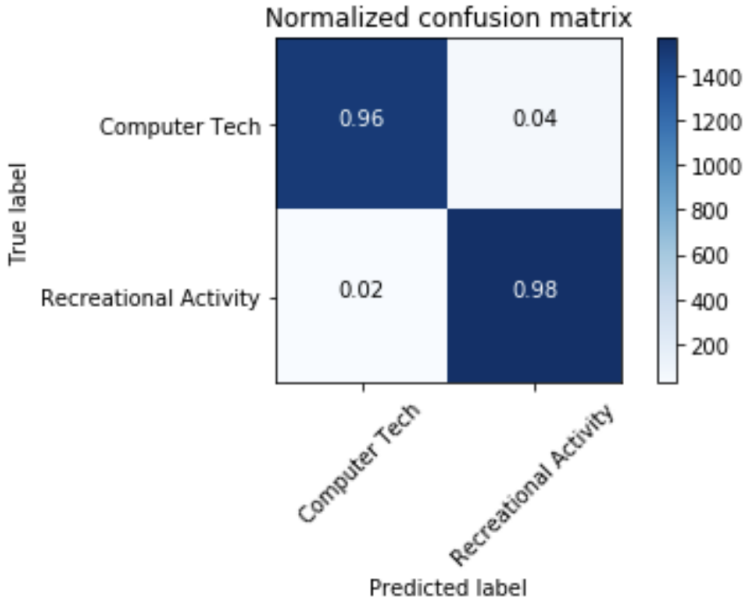


Figure 34: Normalized Confusion Matrix(min_df=2, LSI)

2. LSI with min_df = 5

```
Accuracy of LSI is 0.970476190476
Precision of LSI is 0.962894248609
Recall of LSI is 0.979245283019
```

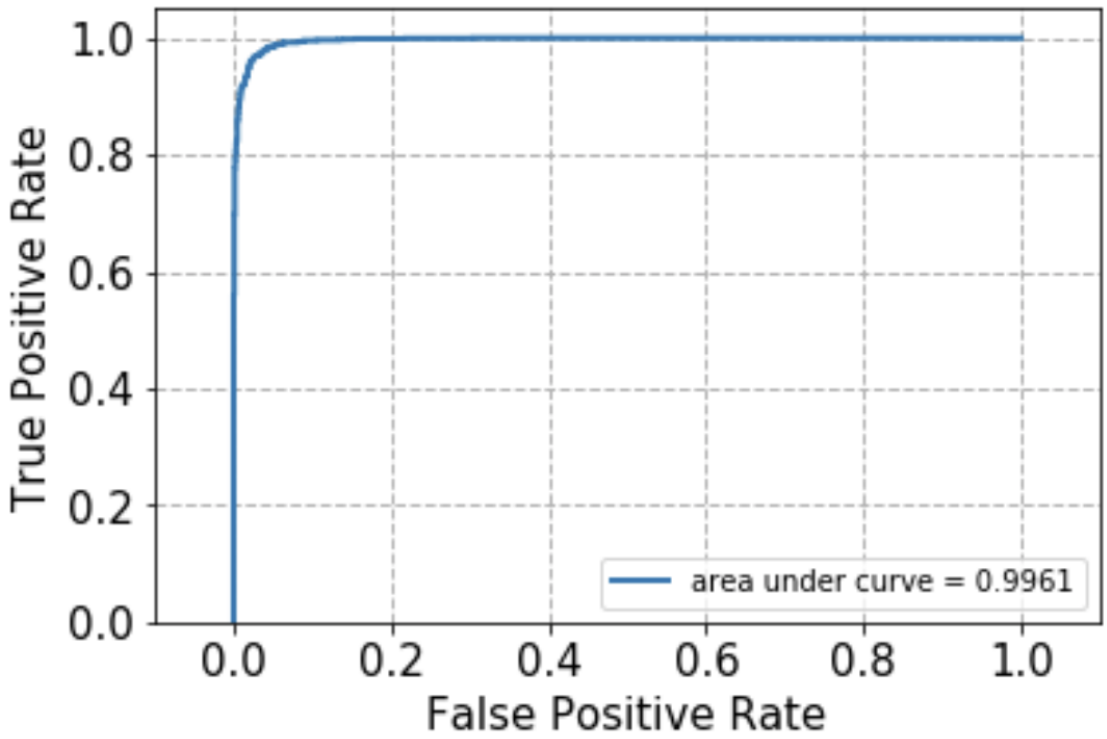


Figure 35: ROC Curve(min_df=5, LSI)

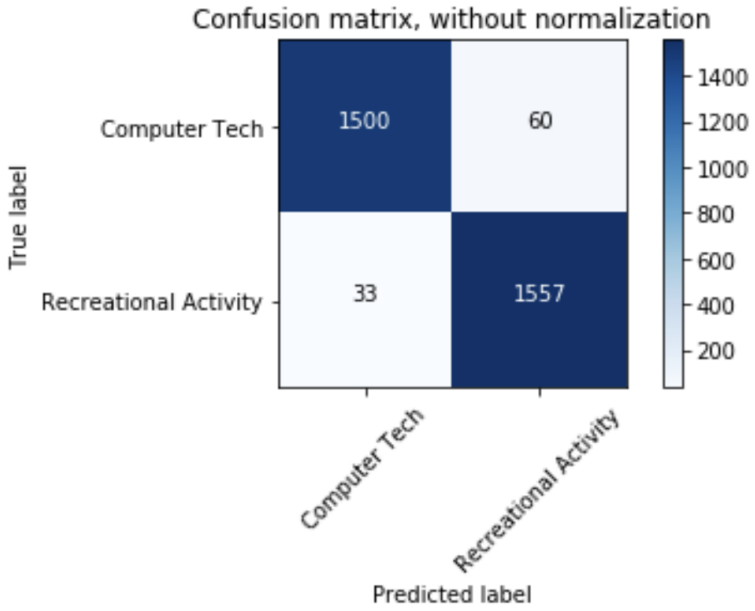


Figure 36: Unnormalized Confusion Matrix(min_df=5, LSI)

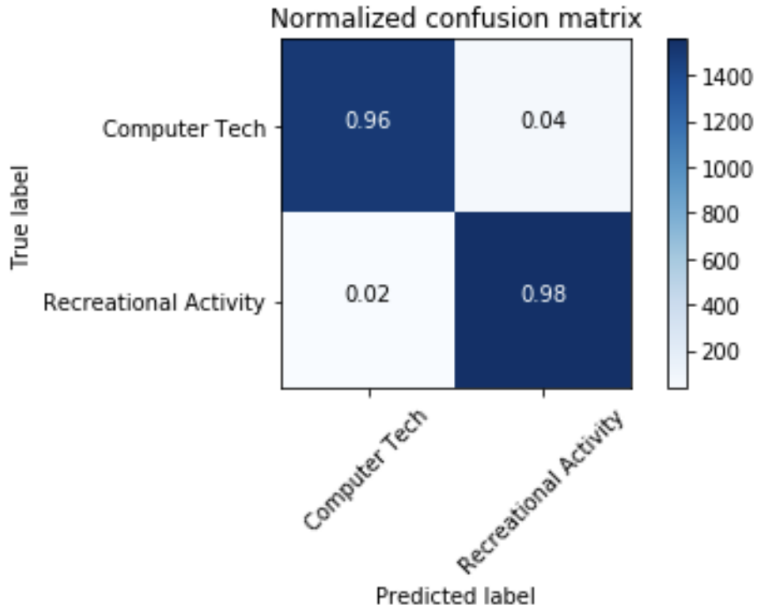


Figure 37: Normalized Confusion Matrix(min_df=5, LSI)

3. NMF with min_df = 2

```
Accuracy of NMF is 0.967301587302
Precision of NMF is 0.954740061162
Recall of NMF is 0.981761006289
```

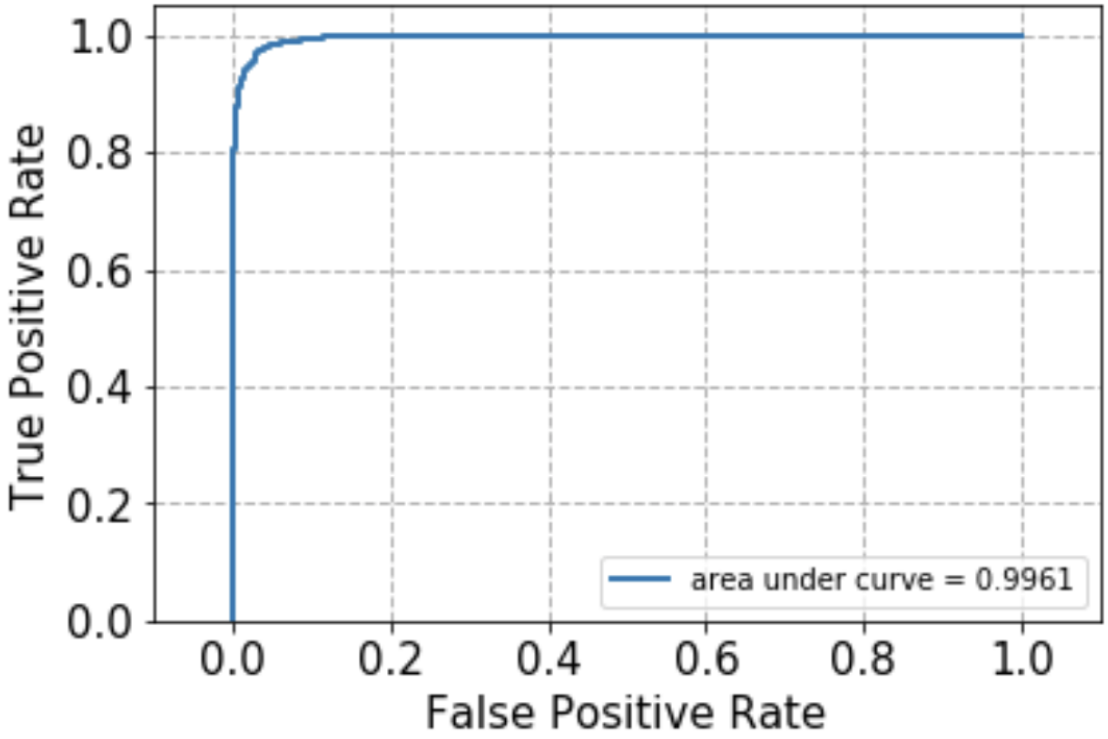


Figure 38: ROC Curve(min_df=2, NMF)

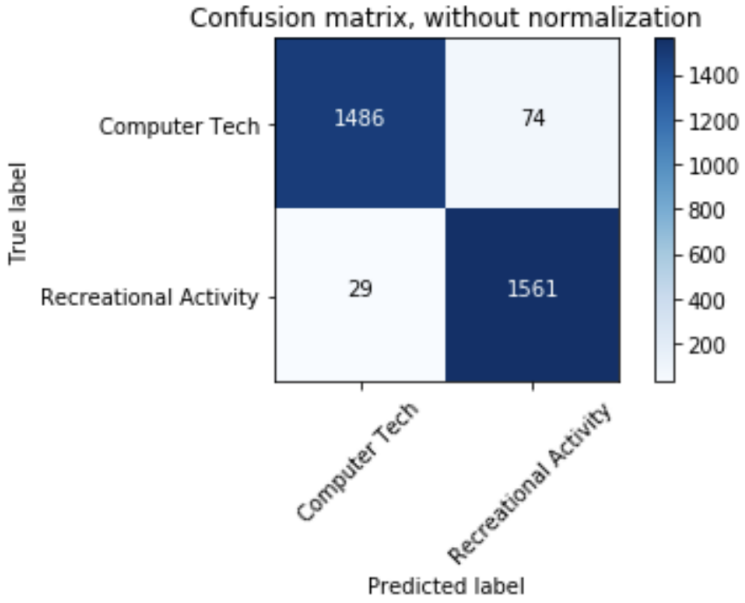


Figure 39: Unnormalized Confusion Matrix(min_df=2, NMF)

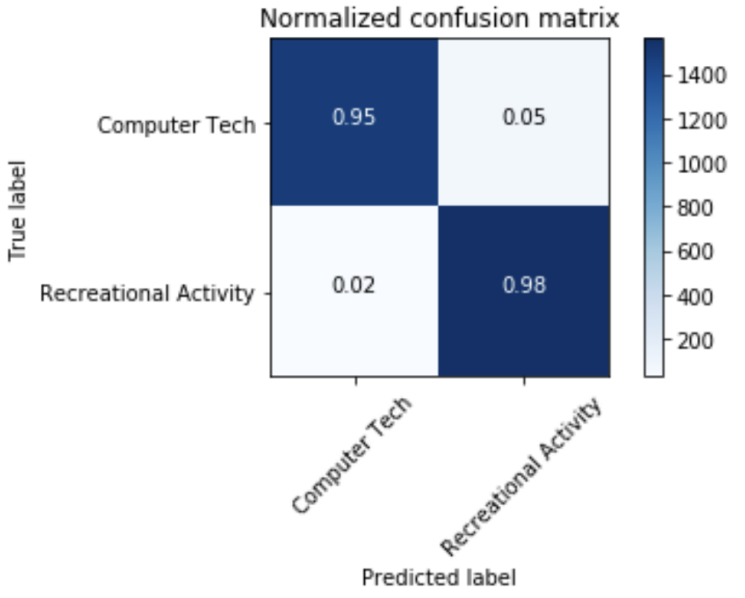


Figure 40: Normalized Confusion Matrix(min_df=2, NMF)

5.5 Question i: Logistic Regression Classifier with Regularization Terms

We add two norm regularization terms to the optimization objective in this part and repeat the processes in question (h). The experiment results are shown in the following figures. According to the results, we can find that the best regularization parameters are generally moderate in scale, neither too large nor too small, which means the test accuracy increases with the regularization parameter first and decreases after a certain point. For the classifiers in this part, norm regularization l1 performs better than l2. Then, as for l1 norm regularizationton, it is obvious that there are more zeros in the coefficients of hyperplane while the coefficients under l2 norm regularization are more evenly. Hence, we should be interested in each type of regularization. If we want to focus on and emphasize some specific features, we use l1 term regularization. Otherwise, if we want each feature to contribute more evenly, we choose l2.

5.5.1 Norm Regularization l1

1. LSI with min_df = 2

```
Regularization parameter γ is 1
Hyperplane coefficients are:  [[ -2.62 71.67 25.42 -25.46 0.  -7.28 10.95
0.  12.07 0.  -6.76 11.45 -2.74 3.89 0.  -4.  -1.27 4.87 -0.64 0.  -2.97
2.93 0.  0.  4.41 0.  0.  0.  3.53 0.  0.  0.  0.  -1.32 -2.88 0.  0.  0.
0.  0.  0.  0.  -0.28 0.  0.  0.  0.  0.  0.  0.  ]]
Accuracy of LSI l1 is 0.967619047619
Precision of LSI l1 is 0.95869297164
Recall of LSI l1 is 0.977987421384
```

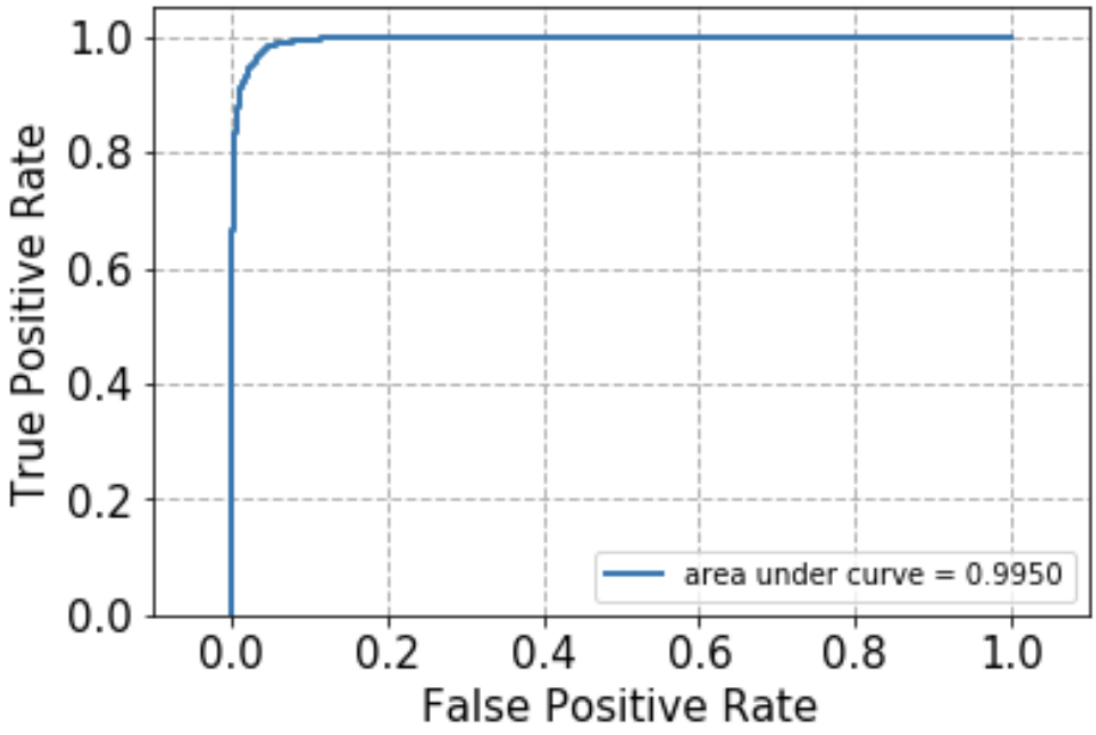


Figure 41: ROC Curve($\gamma = 1$, min_df=2, LSI)

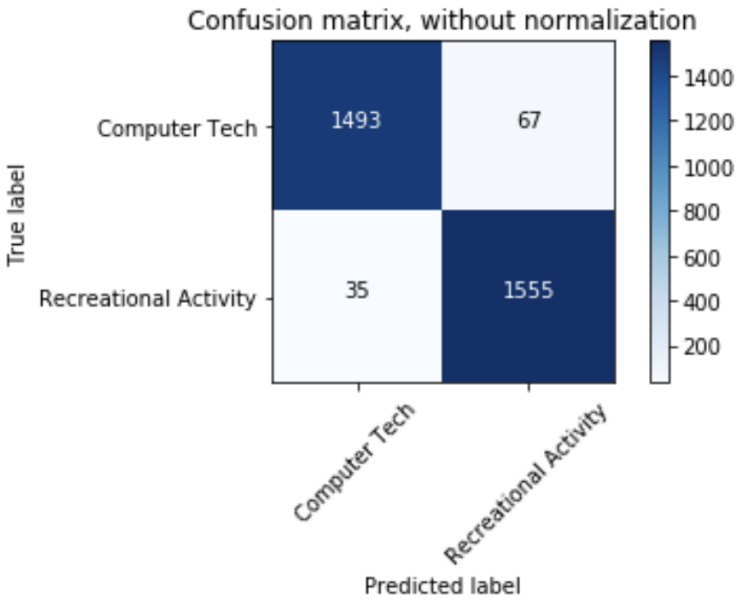


Figure 42: Unnormalized Confusion Matrix($\gamma = 1$, min_df=2, LSI)

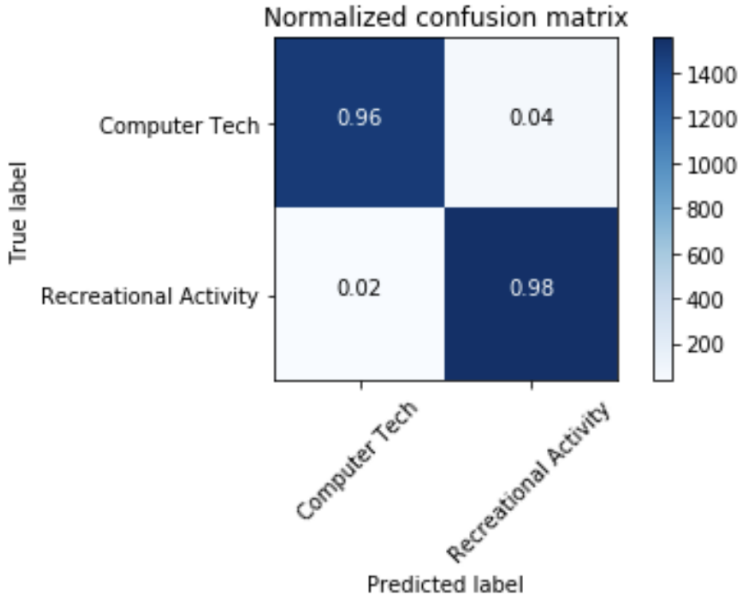


Figure 43: Normalized Confusion Matrix($\gamma = 1$, min_df=2, LSI)

2. LSI with min_df = 5

```
Regularization parameter γ is 1
Hyperplane coefficients are:  [[ -3.14 69.25 23.67 -22.21 -0.17 -12.48 2.96
0.   9.3 9.41 -4.02 10.98 3.12 0.   0.   1.6 0.   1.44 -0.31 -4.22 4.47 0.   0.
-1.17 2.44 5.81 1.73 0.   0.   0.   0.   0.   0.   -6.06 0.   0.   0.   0.   0.   0.
0.   0.   0.1 -1.19 0.41 0.   0.   0.   0.   0.   ]]
Accuracy of LSI l1 is 0.967619047619
Precision of LSI l1 is 0.95869297164
Recall of LSI l1 is 0.977987421384
```

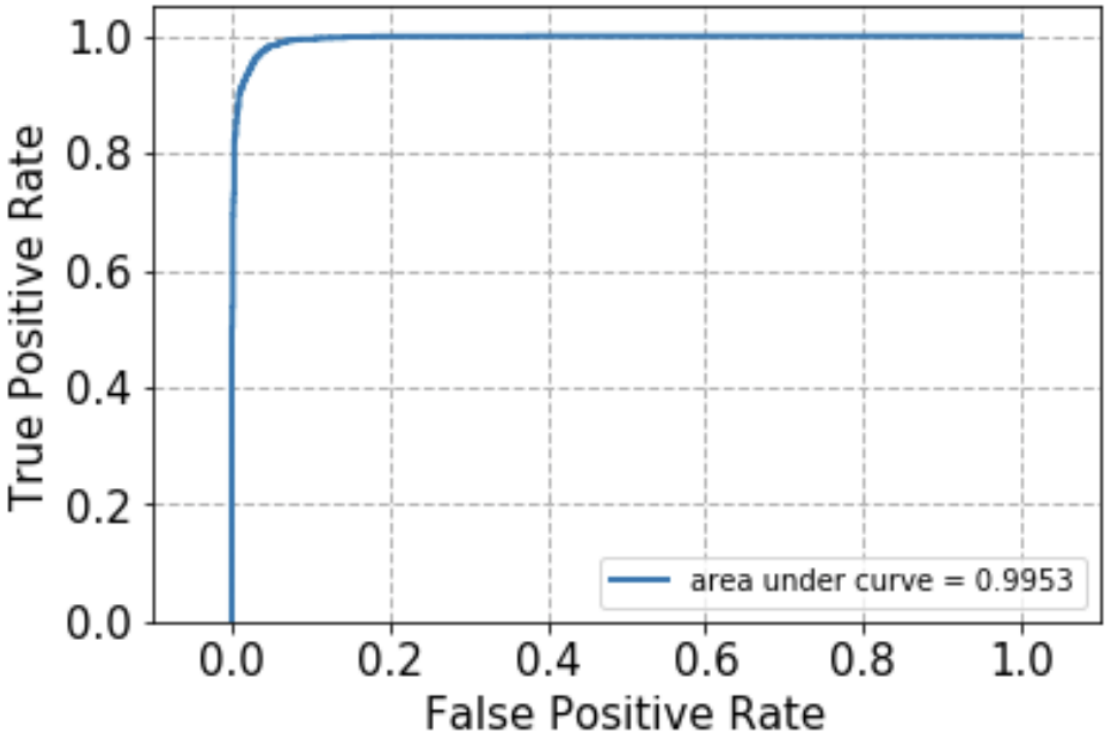


Figure 44: ROC Curve($\gamma = 1$, min_df=5, LSI)

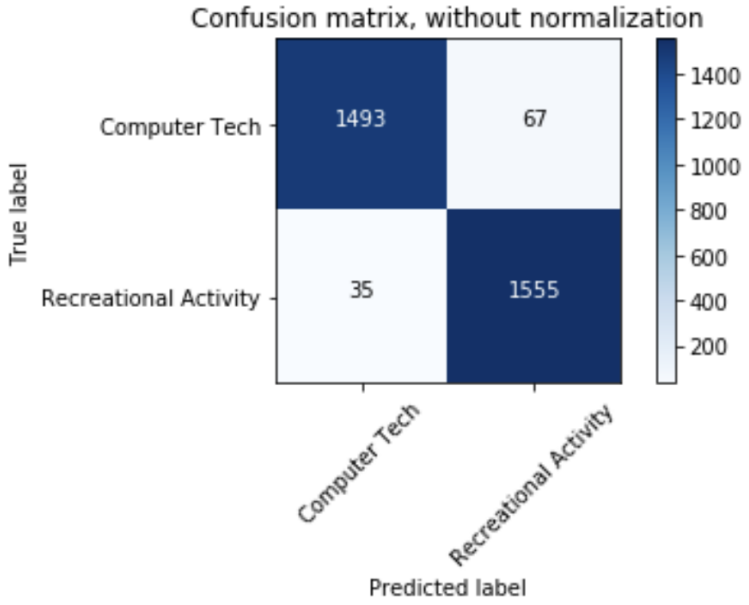


Figure 45: Unnormalized Confusion Matrix($\gamma = 1$, min_df=5, LSI)

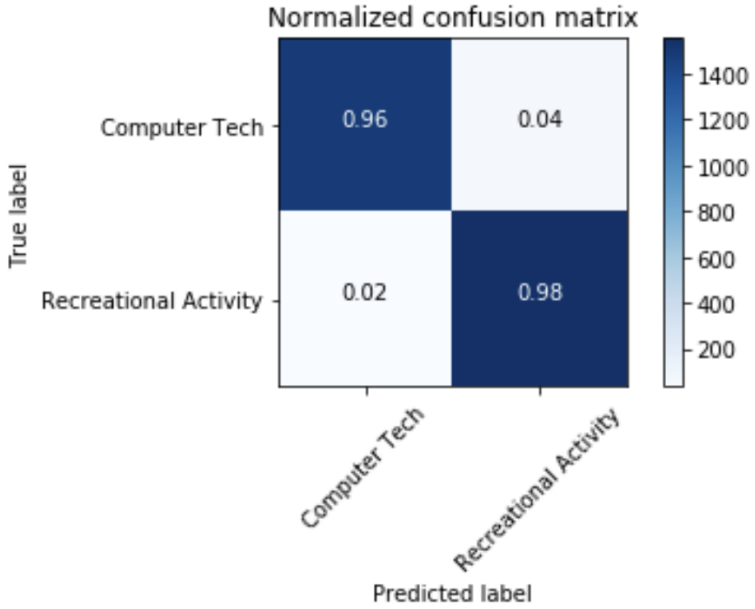


Figure 46: Normalized Confusion Matrix($\gamma = 1$, min_df=5, LSI)

3. NMF with min_df = 2

```
Regularization parameter γ is 10
Hyperplane coefficients are:  [[ 7.6 -5.09 0.  0.  0.  0.  0.  0.  0.  0.
0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
0.  0.  ]]
Accuracy of NMF l1 is 0.745396825397
Precision of NMF l1 is 0.780626780627
Recall of NMF l1 is 0.689308176101
```

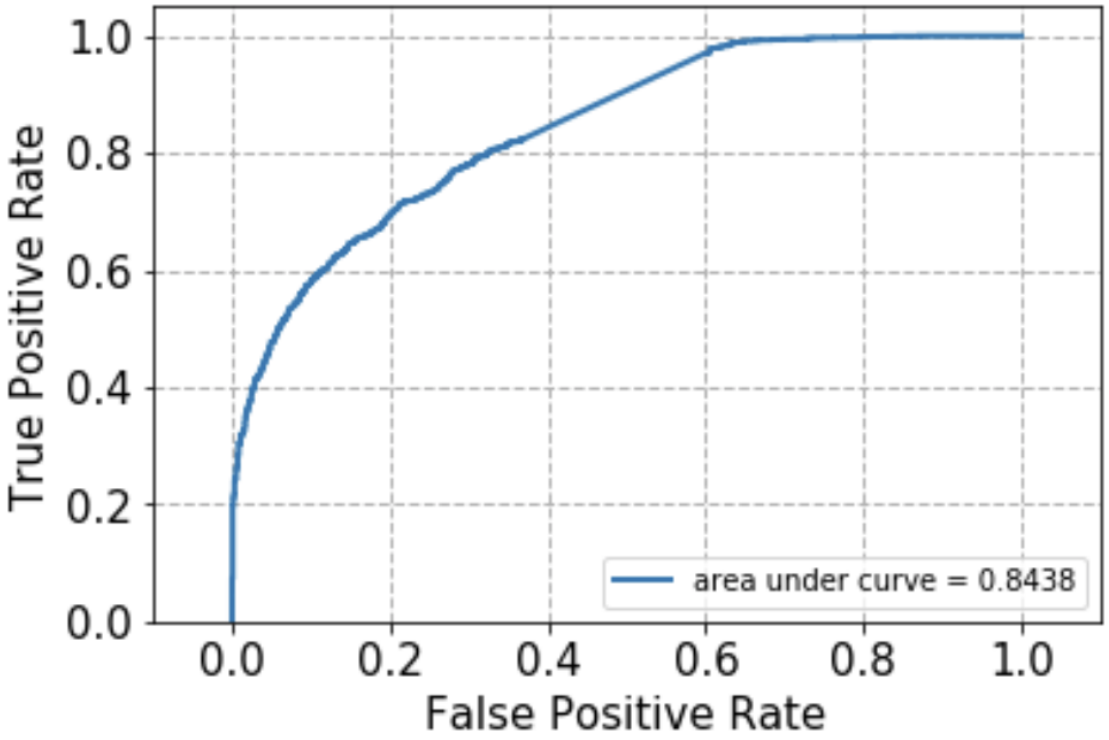


Figure 47: ROC Curve($\gamma = 10$, min_df=2, NMF)

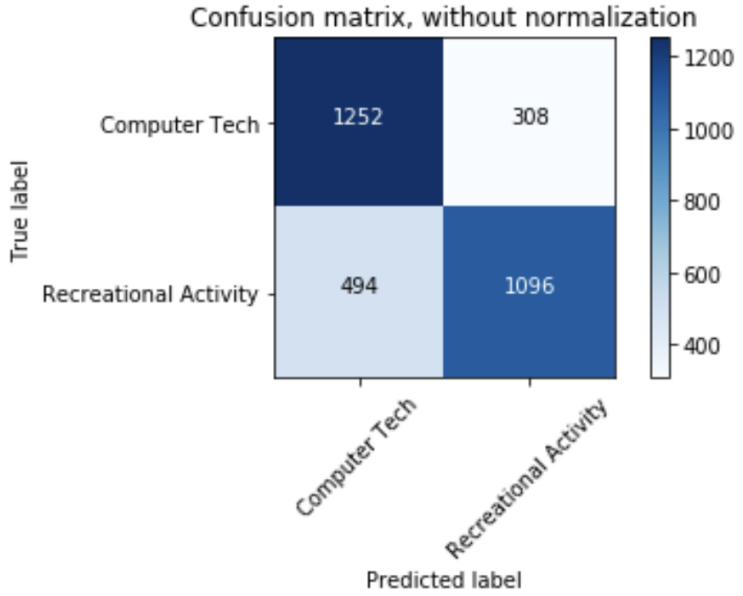


Figure 48: Unnormalized Confusion Matrix($\gamma = 10$, min_df=2, NMF)

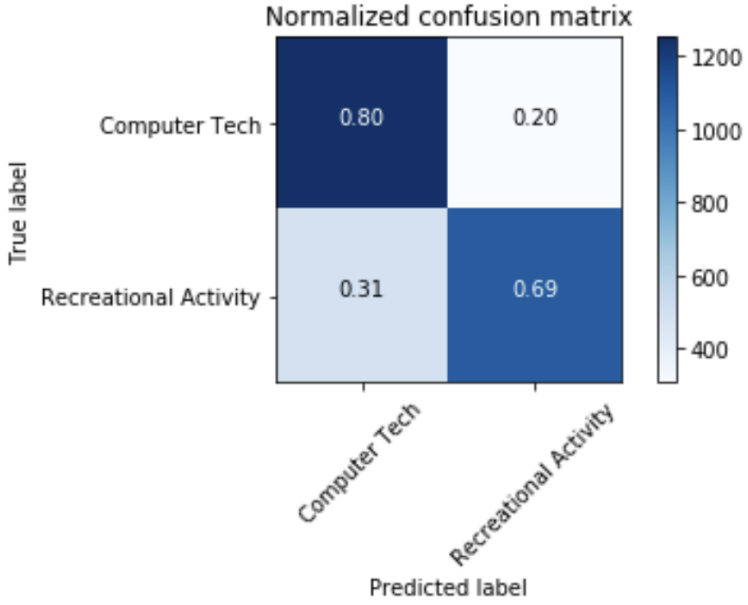


Figure 49: Normalized Confusion Matrix($\gamma = 10$, min_df=2, NMF)

5.5.2 Norm Regularization l2

1. LSI with min_df = 2

```
Regularization parameter γ is 10
Hyperplane coefficients are:  [[ -5.01e-01 8.35e+00 2.85e+00 -2.47e+00 -7.23e-02
-4.00e-01 1.65e+00 -3.21e-02 7.34e-01 1.81e-01 -6.28e-01 2.49e-01 -3.71e-01
-1.02e-01 5.66e-01 1.50e-02 -2.77e-01 5.32e-01 -3.12e-01 5.42e-02 -5.44e-01
5.30e-01 -1.50e-02 1.25e-01 3.73e-01 1.53e-01 3.91e-01 -6.51e-02 2.72e-01
-6.76e-02 -8.74e-02 -2.86e-01 -4.03e-02 -2.83e-01 -3.23e-01 -1.66e-01 -3.46e-02
-1.61e-01 -2.17e-02 2.95e-01 1.11e-01 9.94e-02 -3.46e-01 9.69e-02 -1.84e-01
-1.56e-01 2.96e-03 -5.62e-02 -8.34e-02 1.43e-01]]
Accuracy of LSI l2 is 0.954920634921
Precision of LSI l2 is 0.933532934132
Recall of LSI l2 is 0.980503144654
```

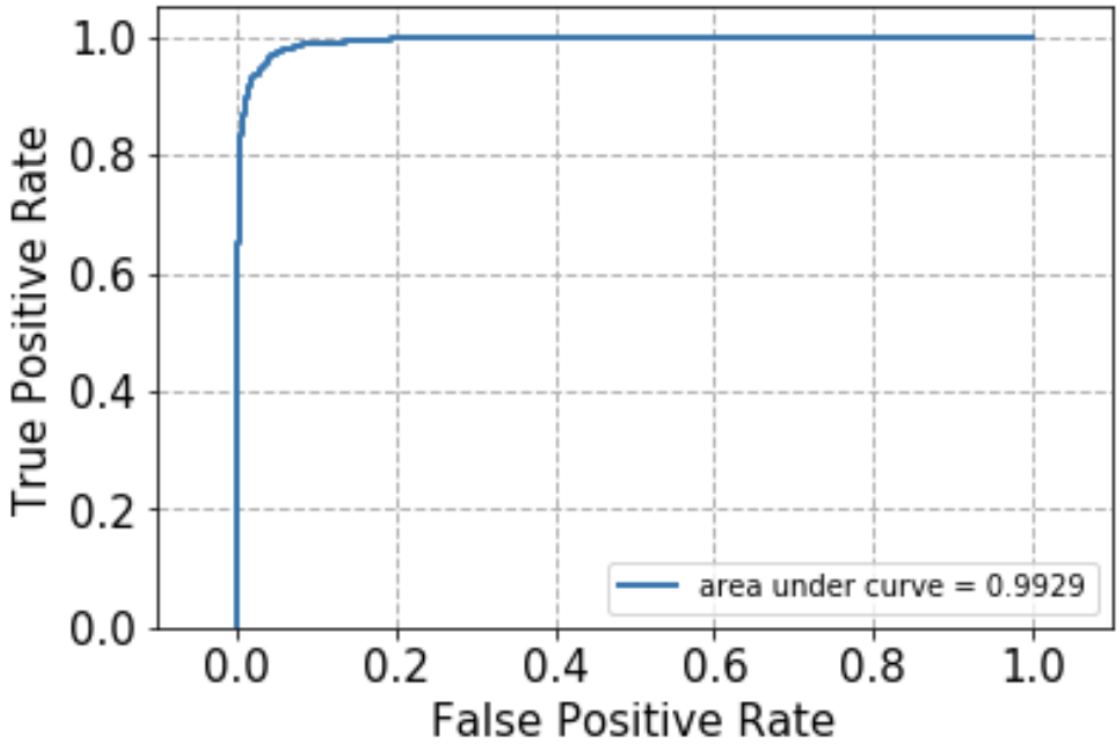


Figure 50: ROC Curve($\gamma = 10$, min_df=2, LSI)

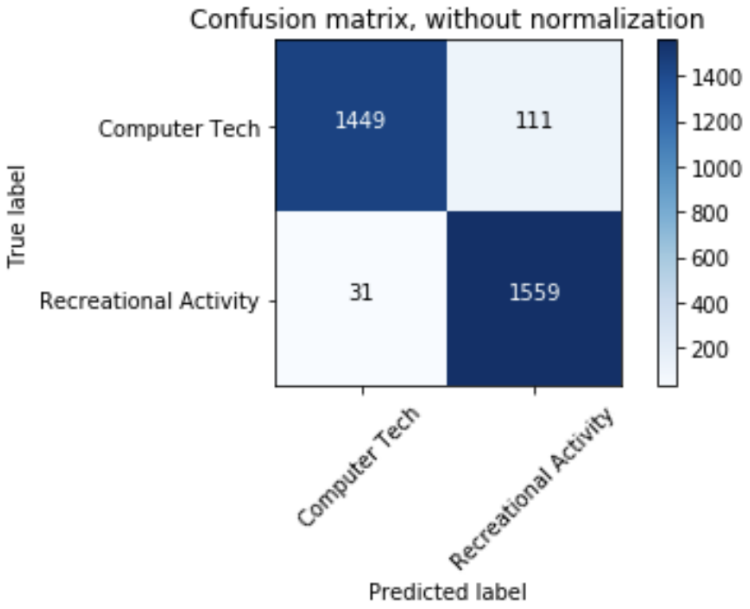


Figure 51: Unnormalized Confusion Matrix($\gamma = 10$, min_df=2, LSI)

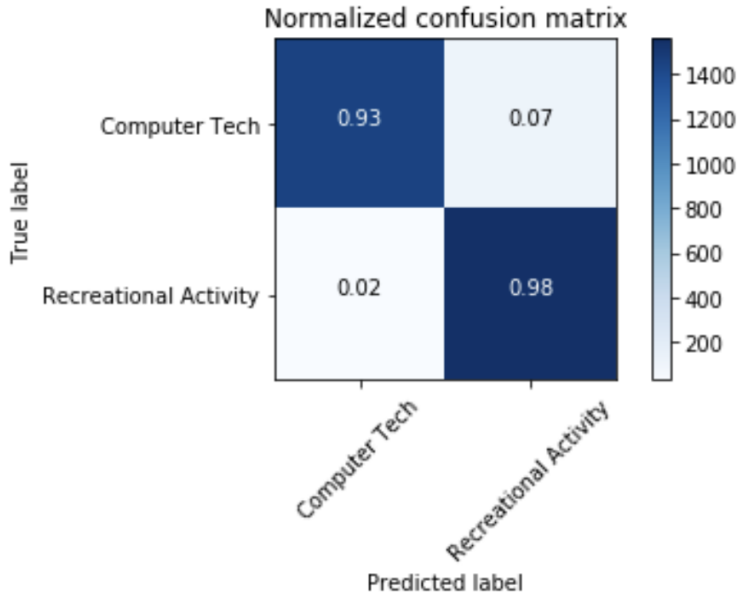


Figure 52: Normalized Confusion Matrix($\gamma = 10$, min_df=2, LSI)

2. LSI with min_df = 5

```
Regularization parameter γ is 1
Hyperplane coefficients are:  [[ -1.84e+00 2.42e+01 9.50e+00 -7.11e+00 -6.22e-01
-3.55e+00 4.89e+00 2.15e+00 2.81e+00 2.77e+00 -1.97e+00 1.71e+00 1.18e+00
-1.58e+00 -1.83e+00 -1.26e-01 -3.91e-01 2.45e+00 -6.05e-01 -2.30e+00 2.27e+00
-5.85e-01 3.36e-01 -1.54e+00 1.27e+00 1.96e+00 1.78e+00 4.61e-01 2.47e-01
1.05e+00 -6.28e-02 1.05e+00 -6.22e-01 -1.89e+00 -2.63e-01 -1.48e-01 2.48e-02
3.23e-01 1.21e+00 -1.15e+00 -2.95e-03 9.29e-01 1.56e+00 -1.04e+00 1.82e+00
-5.81e-01 -7.32e-01 1.03e+00 5.42e-01 -3.00e-01]]
Accuracy of LSI l2 is 0.967619047619
Precision of LSI l2 is 0.959259259259
Recall of LSI l2 is 0.977358490566
```

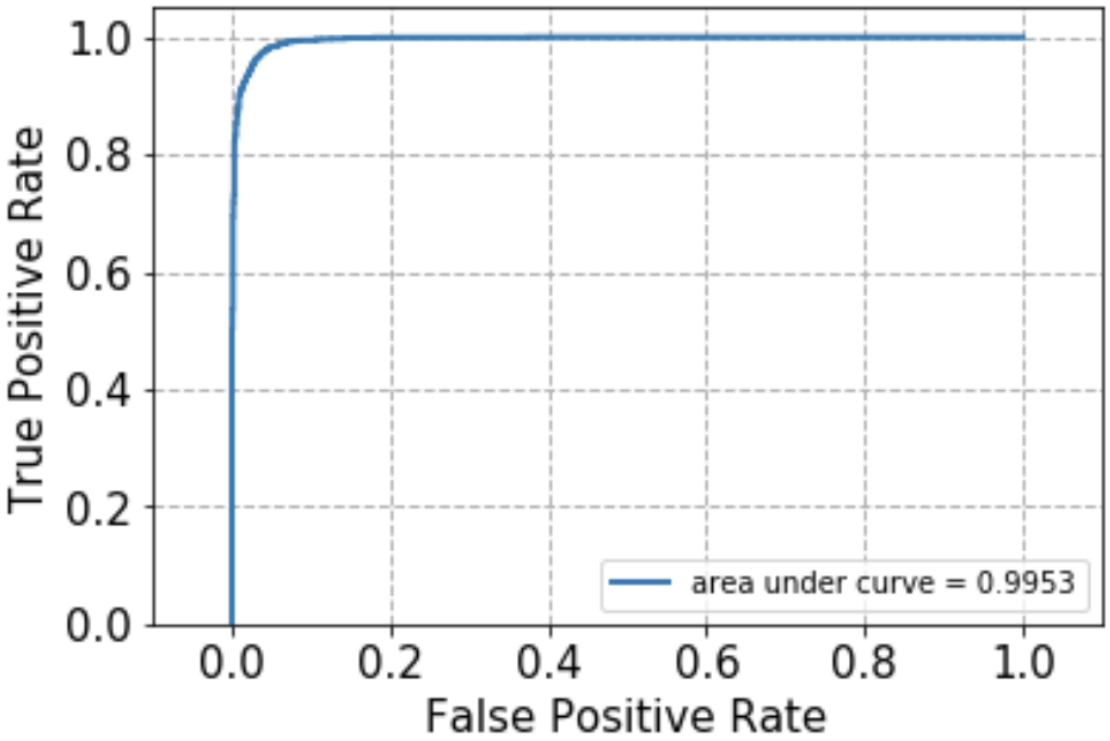


Figure 53: ROC Curve($\gamma = 1$, min_df=5, LSI)

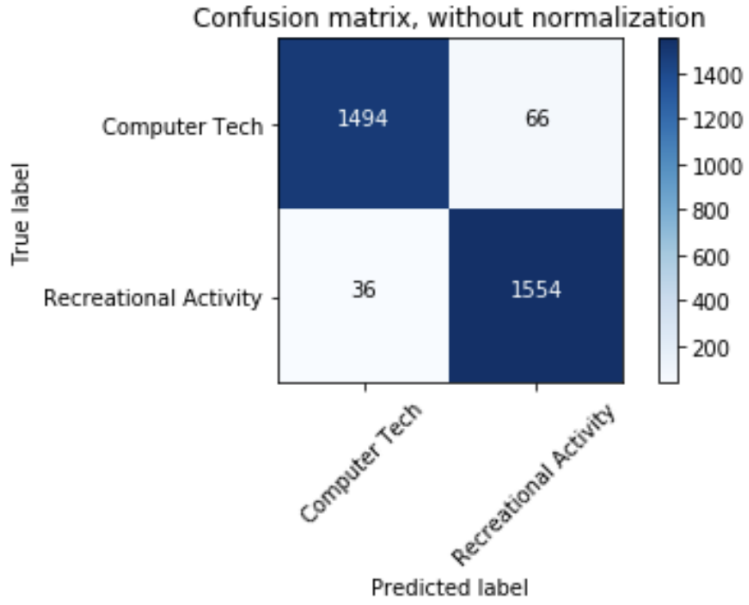


Figure 54: Unnormalized Confusion Matrix($\gamma = 1$, min_df=5, LSI)

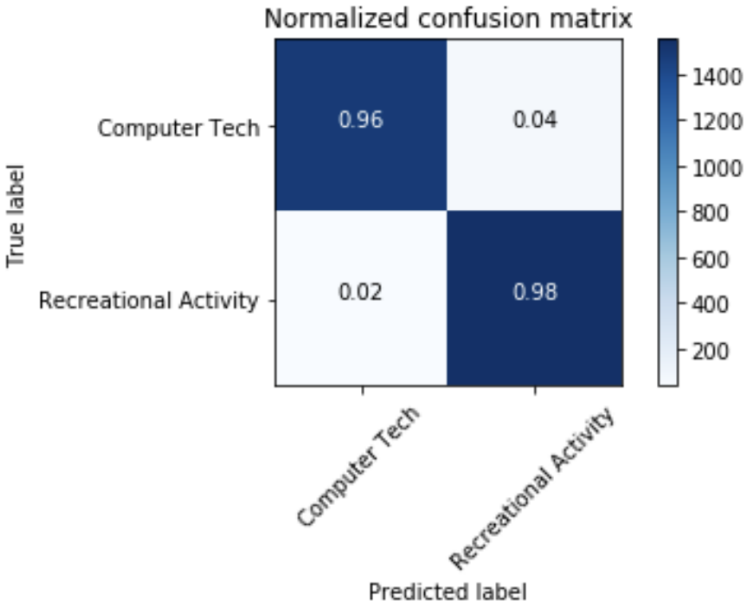


Figure 55: Normalized Confusion Matrix($\gamma = 1$, min_df=5, LSI)

3. NMF with min_df = 2

```
Regularization parameter γ is 100
Hyperplane coefficients are:  [[ 0.14 -0.17 -0.03 -0.09 0.05 0.08 -0.07
0.04 -0.06 0.06 -0.05 0.05 -0.04 -0.02 -0.06 -0.04 0.02 0.02 0.04 -0.04
0.04 0.  -0.07 0.02 -0.03 -0.03 0.02 0.04 0.07 -0.06 0.  0.03 0.02 -0.09
0.02 -0.02 -0.03 -0.03 -0.08 0.04 0.05 0.03 -0.06 0.04 0.03 0.03 0.03 -0.04
-0.03 -0.04]]
Accuracy of NMF l2 is 0.529206349206
Precision of NMF l2 is 0.517409697364
Recall of NMF l2 is 1.0
```

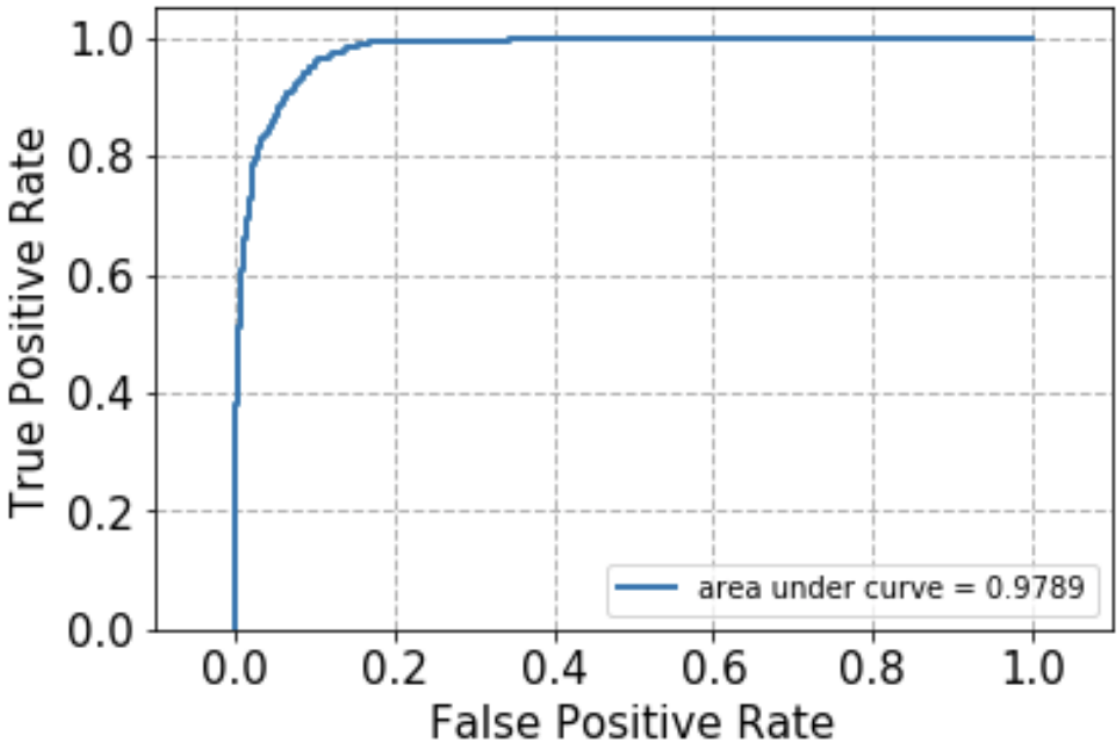


Figure 56: ROC Curve($\gamma = 100$, min_df=2, NMF)

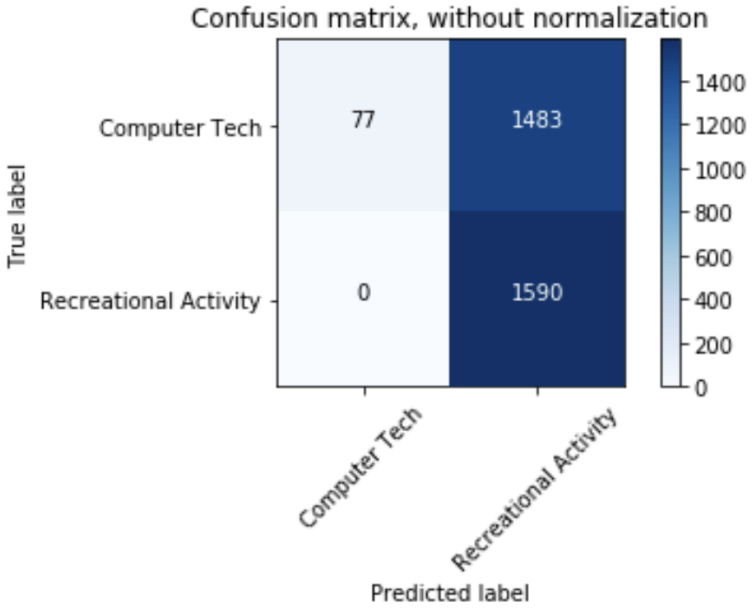


Figure 57: Unnormalized Confusion Matrix($\gamma = 100$, min_df=2, NMF)

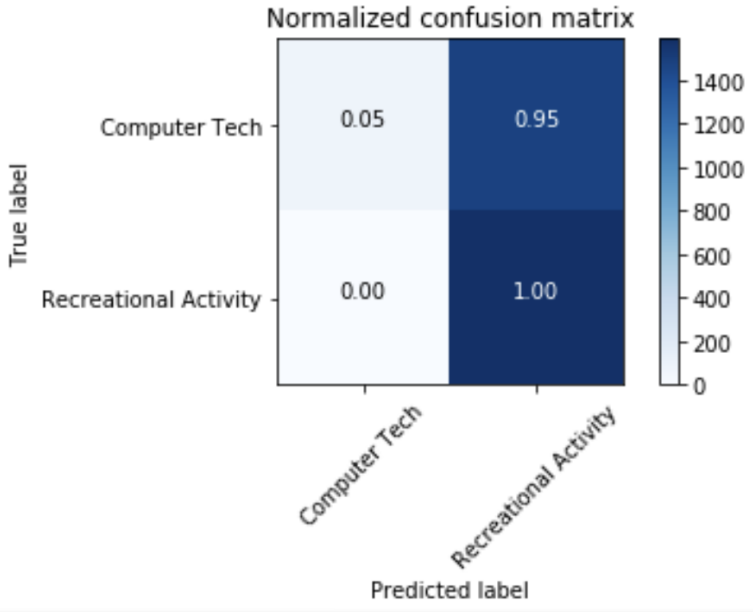


Figure 58: Normalized Confusion Matrix($\gamma = 100$, min_df=2, NMF)

6 Multiclass Classification

6.1 Question j: Multiclass Classification with Naïve Bayes and SVM

For the last question, we need to use Naïve Bayes and SVM algorithm to classify documents under four classes. The same evaluation measures on these classifiers are also listed below.

6.1.1 Naïve Bayes Classifier

1. NMF with min_df = 2

```
Accuracy of NB NMF is 0.798083067093
Recall of NB NMF is 0.798083067093
Precision of NB NMF is 0.812493176984
```

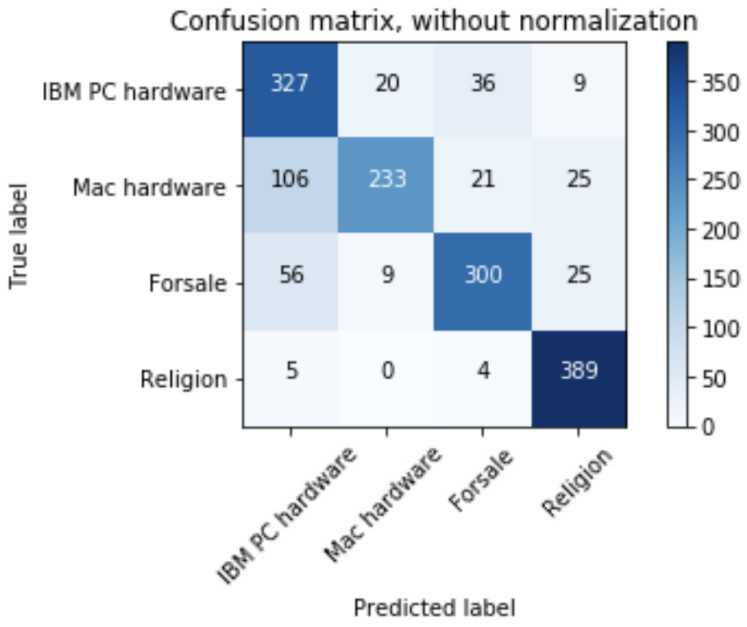


Figure 59: Confusion Matrix Without Normalization(min_df=2, NMF)

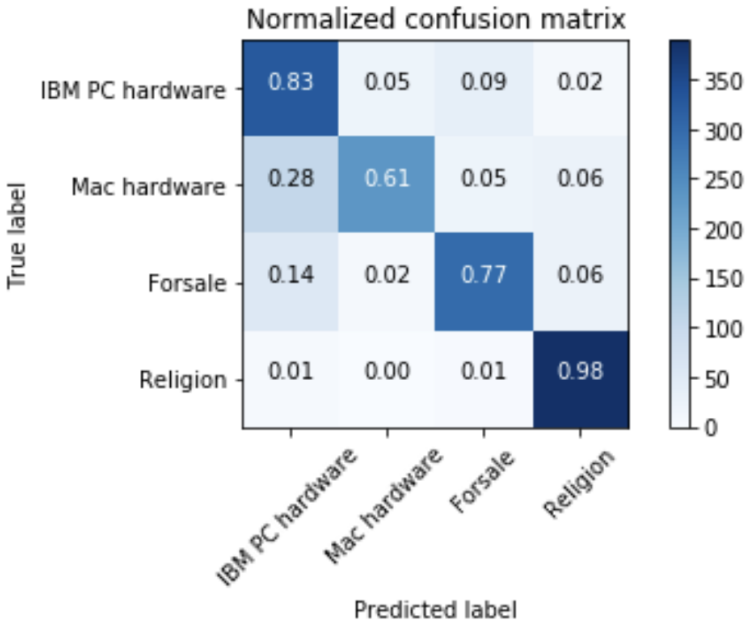


Figure 60: Normalized Confusion Matrix(min_df=2, NMF)

6.1.2 SVM Classifier

1. One VS One LSI with min_df = 2

```
Accuracy of SVM One vs One LSI is 0.876038338658
Recall of SVM One vs One LSI is 0.876038338658
Precision of SVM One vs One LSI is 0.877825434676
```

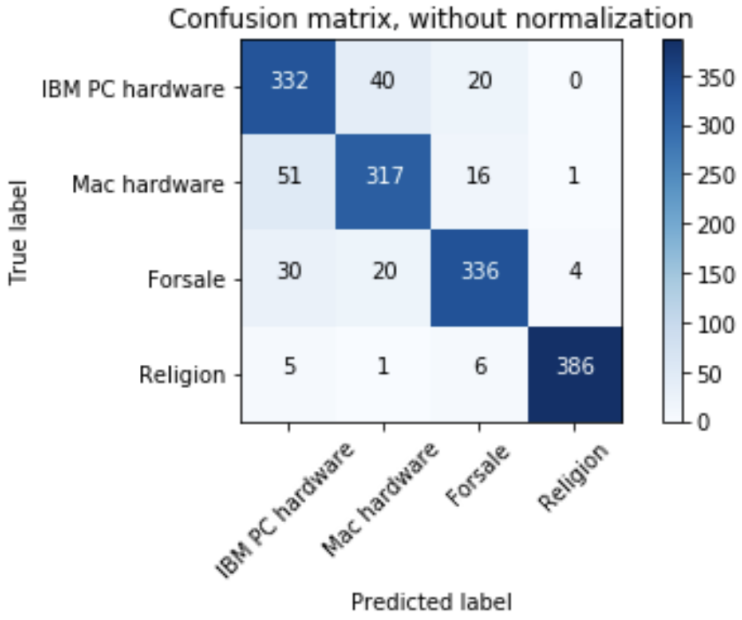


Figure 61: Confusion Matrix Without Normalization(min_df=2, LSI, One VS One)

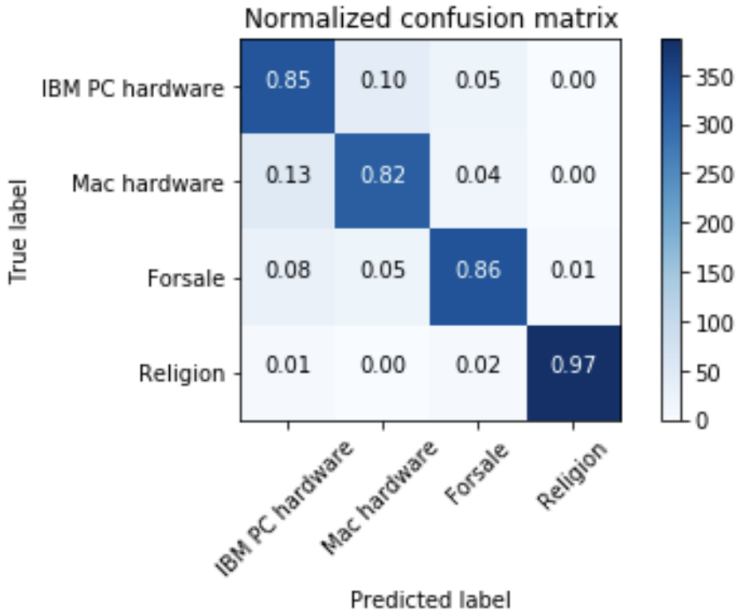


Figure 62: Normalized Confusion Matrix(min_df=2, LSI, One VS One)

2. One VS Rest LSI with min_df = 2
   `Accuracy of SVM One vs Rest LSI is 0.879872204473`
   `Recall of SVM One vs Rest LSI is 0.879872204473`
   `Precision of SVM One vs Rest LSI is 0.880826635121`

3. One VS One NMF with min_df = 2
   `Accuracy of SVM One vs One NMF is 0.849840255591`
   `Recall of SVM One vs One NMF is 0.849840255591`
   `Precision of SVM One vs One NMF is 0.852728967755`

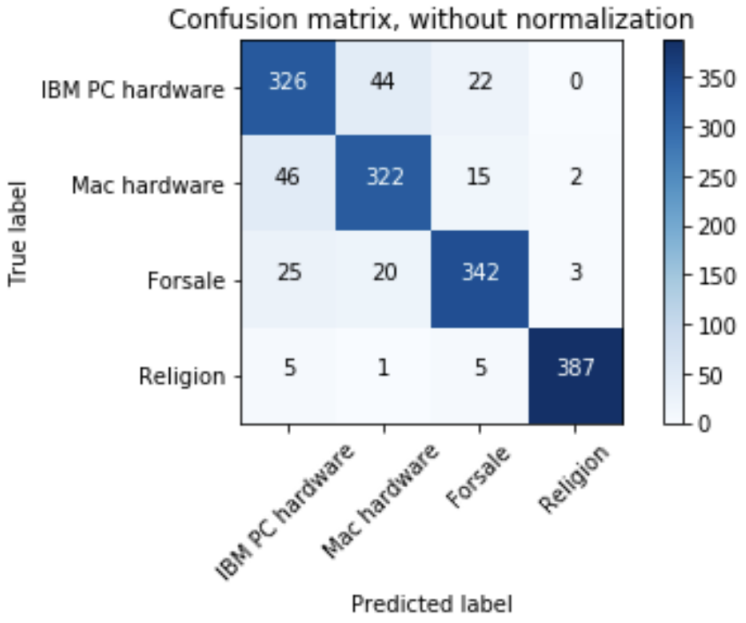


Figure 63: Confusion Matrix Without Normalization(min_df=2, LSI, One VS Rest)

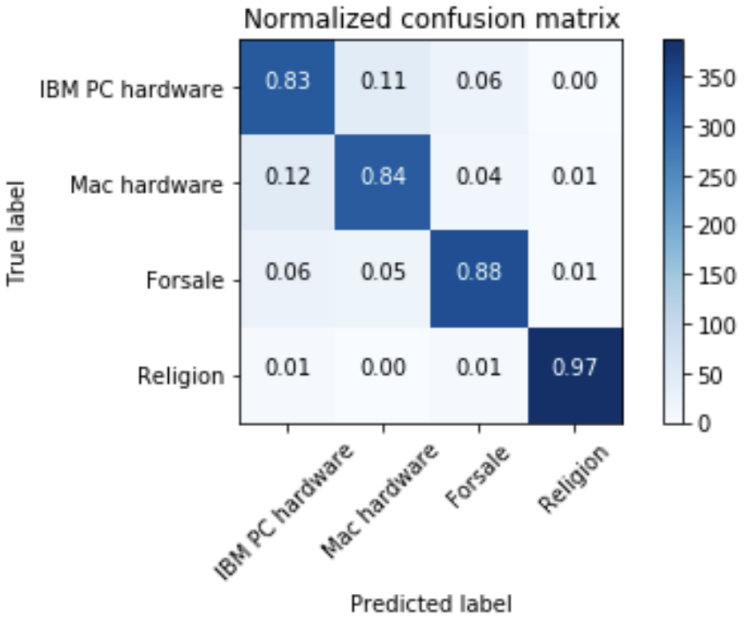


Figure 64: Normalized Confusion Matrix(min_df=2, LSI, One VS Rest)

4. One VS Rest NMF with min_df = 2

```
Accuracy of SVM One vs Rest NMF is 0.853674121406
Recall of SVM One vs Rest NMF is 0.853674121406
Precision of SVM One vs Rest NMF is 0.8539750664
```

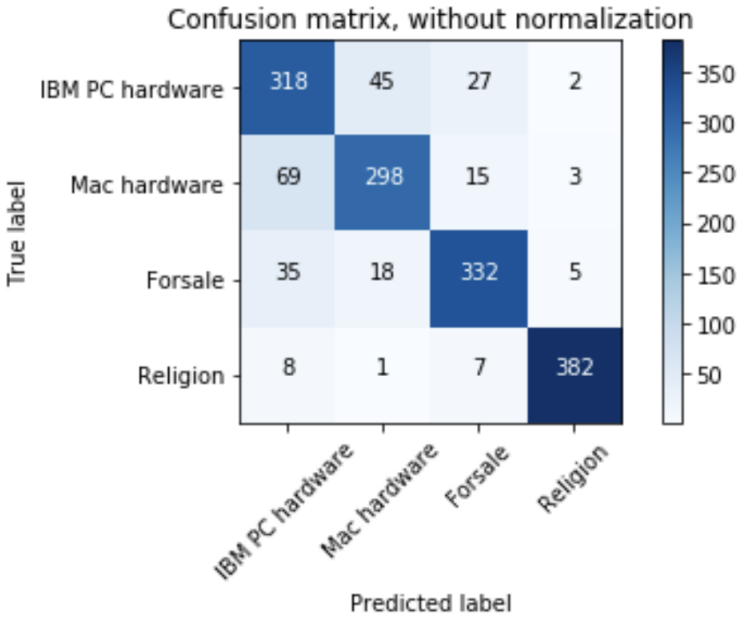


Figure 65: Confusion Matrix Without Normalization(min_df=2, NMF, One VS One)

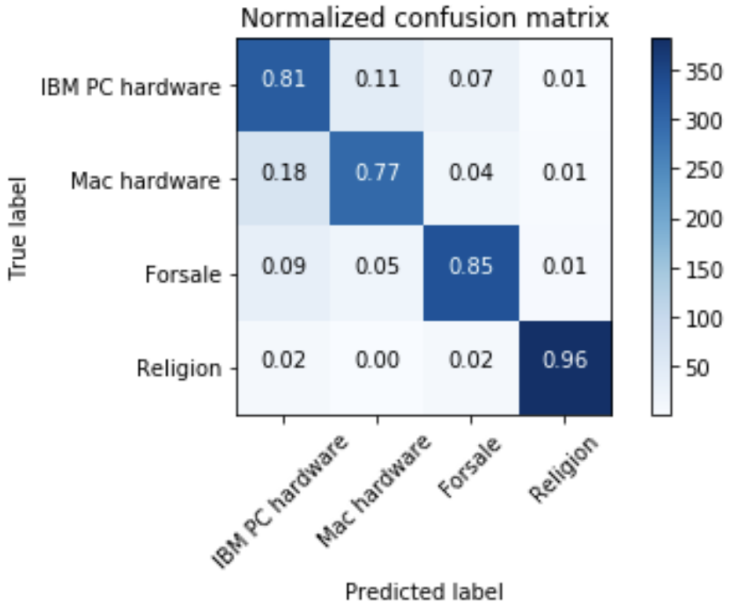


Figure 66: Normalized Confusion Matrix(min_df=2, NMF, One VS One)

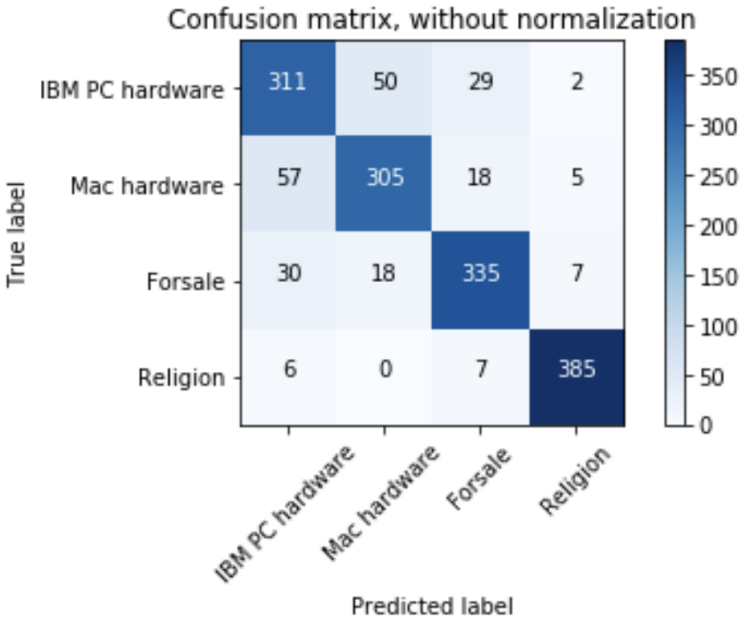


Figure 67: Confusion Matrix Without Normalization(min_df=2, NMF, One VS Rest)

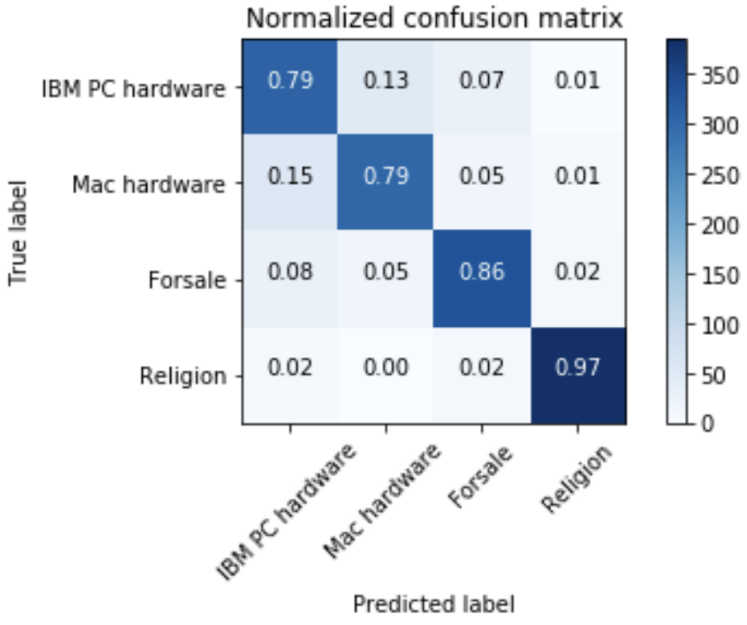


Figure 68: Normalized Confusion Matrix(min_df=2, NMF, One VS Rest)