Project #1

Contributor:

Yuting Tang 304881011

Xiao PENG 005033608

Date:

Jan 29, 2018

Table of Contents

Introduction 1

Dataset and Problem Statement 3

Problem A 4

Modeling Text Data and Feature Extraction 4

Problem B 4

Problem C 6

Feature Selection 6

Problem D 7

Learning Algorithms 7

Problem E 8

Problem F 18

Problem G 22

Problem H 30

Problem I 35

Multiclass Classification 43

Problem J 43

Introduction

The main purpose of this project is to understand and learn the basic procedures of the model evaluation in classification analysis, which is the most significant method to analyze large scale data. In order to achieve this goal, in this project we are required to perform several classification methods to classify relatively large amount of textual data to predefined labels where they belong to according to inherent structure of the features based on '20 Newsgroups dataset'. Firstly, we fetch and preprocess the textual data by creating the TFxIDF vector representation and dimension reduction so that the data can be fitted into the training model. After that, we apply different kinds of classifiers, like linear SVM (Support Vector Machine), multinomial Naive Bayes and Logistic Regression, with different values of parameters. By processing the data with different classifiers, we calculate recall, ROC curve, accuracy, precision and F1-score from the confusion metrics to evaluate the performance of the classifiers.

Through this project, we gain deeper understanding of the inner mechanism of preprocessing methods for textual data and classification algorithm.

Dataset and Problem Statement

"20 Newsgroups" dataset is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups, each corresponding to a different topic. For the purposes, grouping the documents into two classes: Computer Technology and Recreational activity, we work with 8 subclasses as shown in Table 1.

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

Table 1: Subclasses of 'Computer technology' and 'Recreational activity'

Problem A

Requirement：plot a histogram of the number of training documents per class to check if they are evenly distributed.

Result: We load the datasets by importing the function 'fetch_20newsgroups', changing the 'category' parameter. Attached behind is our results of this problem, the histogram of the number of documents per topic in the whole '20 Newsgroups' dataset. Another purpose is to make sure the data is evenly distributed. From the histogram, we can read the amount of documents in each topics, which is suitable for the process afterwards.

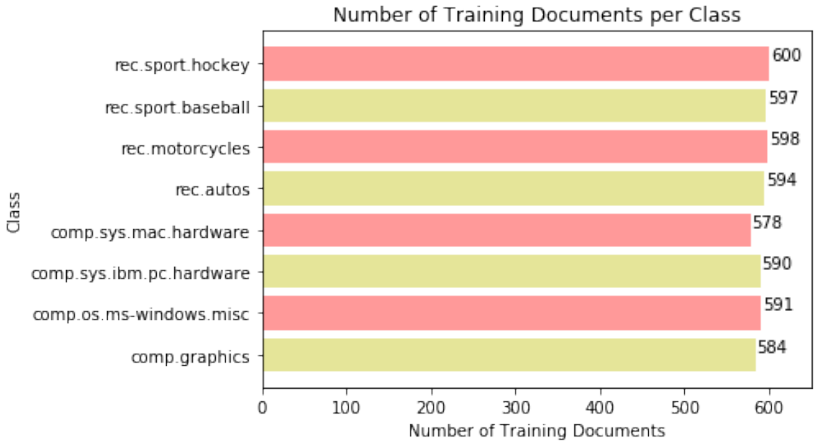


Figure 1: The histogram of document amount per topic

Modeling Text Data and Feature Extraction

Problem B

Requirement: First tokenize each document into words. Then, excluding the stop words, punctuations, and using stemmed version of words, create a TFxIDF vector representations. Use two settings for minimum document frequency of vocabulary terms, namely min_df=2 and min_df=5. Report the final number of terms you extracted.

In order to extract the features of the textual data loaded, we should process the data before we feed it to the feature extraction algorithms, since the raw data, namely a sequence of symbols, are not acceptable to the algorithms, which requires the input to be numerical

feature vectors. For this reason, CountVectorizer whose function is to convert text into a matrix of token counts, is applied to obtain a term-document matrix, the element of which in i^{th} row and j^{th} column represents the frequency of j^{th} term in i^{th} document.

Another significant thing is that we should exclude the stop-words, punctuations and different stems of a word while we transform the textual data into vectors. We define our stem_tokenizer function to tokenize the data then extract the stem form of the words, which is implemented by using Regular expressions filter. After removing all the punctuations and symbols, we call SnowballStemmer function to remove the words with the same meaning. The stop-words parameter in CountVectorizer set to 'english' realize the function to remove the stop words. The min_df parameter can eliminate the words with very low occurring frequency. In other words, when we set min_df as 5, the words occur less than 5 times are removed.

Finally, we call TFIDF (Term Frequency Inverse Document Frequency) metric to balance the significance of those terms appear almost in every document since the metric takes both the repeatability and uniqueness of terms into consideration at the same time.

Result: After all these work, we can output the number of the terms extracted with min_df of 2 and 5 respectively, as shown in Figure 2. When min_df is set as 2, there are 4732 training documents in total and we extract 19385 terms from these documents. When min_df is set as 5, there are 4732 training documents in total and we extract 13813 terms from these documents. When min_df is set as 2, there are 3150 testing documents in total and we extract 8976 terms from these documents. When min_df is set as 5, there are 3150 testing documents in total and we extract 6695 terms from these documents.

```
Dimensions of TF-IDF Vector of the trainning data with min_df=2: (4732, 19385)
Number of Terms Extracted with min_df=2: 19385
Dimensions of TF-IDF Vector of the testing data with min_df=2: (3150, 13813)
Number of Terms Extracted with min_df=2: 13813
Dimensions of TF-IDF Vector of the trainning data with min_df=5: (4732, 8976)
Number of Terms Extracted with min_df=5: 8976
Dimensions of TF-IDF Vector of the testing data with min_df=5: (3150, 6695)
Number of Terms Extracted with min_df=5: 6695
```

Figure 2: The number of terms extracted with min_df of 2 and 5

Problem C

Requirement: find the 10 most significant terms in each of the following classes with respect to TFxICF measure. Note that in this part of your assignment, n_{classes} is 20.

comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, misc.forsale, soc.religion.christian

In order to quantify how significant a word is to a certain class, we can define a measure called TFxICF. Instead of adopting TFxIDF metric modeling in problem b, we use TFxICF to quantify the significance of each term in each topic based on the formula given by

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

Where $f_{t,c}$ is the number of occurrences of the term in documents that belong to class c.

Result: In Figure 3, the followings are the top 10 significant terms, namely terms with the 10 highest values evaluated by the TFxICF function, of the given subgroup. Some terms could seem weird due to the Snowball Stemming algorithm, however this improves the accuracy of the model since it eliminates the repeated terms.

```
Most Significant 10 Terms for comp.sys.ibm.pc.hardware
['aspi', 'penev', 'balog', 'scsiha', 'usma', 'husak', 'fasst', 'schaufenbuel', 'korenek', 'latonia']

Most Significant 10 Terms for comp.sys.mac.hardware
['powerbook', 'lciii', 'adb', 'bmug', 'iivx', 'iifx', 'firstclass', 'jartsu', 'macus', 'nodin']

Most Significant 10 Terms for misc.forsale
['liefeld', 'sabretooth', 'hobgoblin', 'uccxkvb', 'kou', 'radley', 'snes', 'koutd', 'keown', 'spiderman']

Most Significant 10 Terms for soc.religion.christian
['clh', 'liturgi', 'kulikauska', 'mmalt', 'caralv', 'monophysit', 'mussack', 'sspx', 'atterlep', 'schismat']
```

Figure 3: The number of terms extracted with min_df of 2 and 5

Feature Selection

Carefully taking the results in Problem B into consideration, we find they actually have a fatal drawback that the TFxIDF vectors have more than seven thousand features for documents of each topics, which makes this kind of representation vectors sparse and low-rank. Besides, the high dimensionality increases the difficulty to classify while reduces the performance of the learning algorithm. In order to convert the sparse vectors to the normal dense ones, we need to turn to the LSI (Latent Semantic Indexing) and NMF (Non-Negative Matrix

Factorization), both of which contribute to finding the optimal representation vectors. In this project, we choose k, the largest dimension of the vectors, as 50. And we use TruncatedSVD from sklearn decomposition package to reduce the dimension of the vectors to 50. Therefore, we get the selected features for our learning algorithms.

Problem D

Requirement: Apply LSI to the TFxIDF matrix corresponding to the 8 classes. and pick k=50; so each document is mapped to a 50-dimensional vector. Alternatively, reduce dimensionality through Non-Negative Matrix Factorization (NMF) and compare the results of the parts e-i using both methods.

Result: In this task, we reduce the dimension with LSI and NMF with min_df at 2 and 5 respectively. At min_df of 2, before the reduction the shape of the training TFxIDF and testing TFxIDF matrix is (4732, 19385) and (3150, 13813), while after applying LSI or NMF, the shapes of the matrices become (4732, 50) and (3150, 50) respectively. Similarly, at min_df of 5, before the reduction the shape of the training TFxIDF and testing TFxIDF matrix is (4732, 8976) and (3150, 6695), while after applying LSI or NMF, the shapes of the matrices become (4732, 50) and (3150, 50) respectively.

Learning Algorithms

In this part of the project, we begin to get in touch with different kinds of real classifiers. But before we start, some evaluation coefficients are essential to be brought out.

• **Confusion matrix:** Table that describes the performance of a classification model Every observation in the testing set is represented in exactly one box. It's a 2x2 matrix because there are 2 response classes. It allows you to calculate a variety of metrics and it's useful for multi-class problems.

• **True Positives (TP) :** Correctly predicted that they belong to Class 0

• **True Negatives (TN):** Correctly predicted that they belong to Class 1

• **False Positives (FP):** Incorrectly predicted that they belong to Class 0

• **False Negatives (FN):** Incorrectly predicted that they belong to Class 1

• **Accuracy:** is defined as (TP + TN)/(TP + TN + FP + FN)

• **Recall:** is defined as TP/(TP + FN)

• **Precision:** is defined as TP/(TP +FP)

•**ROC Curves:** Tells us how sensitivity and specificity are affected by various thresholds, without actually changing the threshold.

•**Area Under the Curve (AUC):** AUC is the percentage of the ROC plot that is underneath the curve. AUC is useful as a single number summary of classifier performance.

Problem E

Requirement: Use hard margin SVM classifier (SVC) by setting γ to a high value, e.g. 1000, 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. Repeat the same procedure for soft margin SVC (set γ to 0.001)

To be specific, our task in this problem is to predict classification of the testing data, then we should use the several coefficients, such as ROC, confusion matrix, accuracy, recall and prediction to evaluate the performance of the classification model.

In this problem, Linear Support Vector Machine is used to train classification model and we test its accuracy by using new data points. SVM is one of the most frequently used supervised learning models in classification and regression problems. It builds a model to divide categories by defining a clear gap.

And in order to observe how much the min_df can affect the prediction results and the influence that two different dimension reduction methods can bring to the performance of the SVC model, we have to run the whole process at min_df of 2 and 5 respectively with LSI and NMF separately. To achieve this goal better, the strategy we take in this problem is building pipelines to stream all the workflows, the mechanism of which achieve all the steps

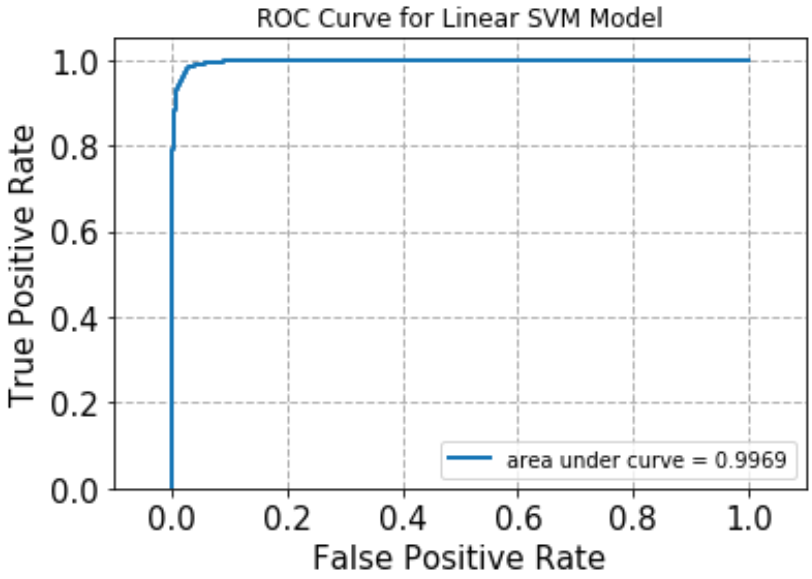
of packaging and management. By using it we can compute the prediction faster and more accurate.

In the pipeline, the first step we take is to transform loaded training data of 8 given topics into the TFxIDF vectors. Secondly, we use LSI and NMF to reduce the dimension of the TFxIDF vectors and apply the linear SVC to train the preprocessed data and predict the classification of the testing data. Finally, we calculate the coefficients and plot the corresponding figure to show the performance of the model straightforward. For hard and soft margin SVC, the procedures are almost identical except we pick the penalty coefficient γ of 1000 and 0.001 respectively.

Result: The results are divided into hard margin SVC with γ of 1000 and soft margin SVC with γ of 0.001.

————————————————RESULTS of hard margin SVC with LSI & min_df = 2

Accuracy: 0. 0.967301587302
Recall: 0. 990566037736
Precision: 0. 947083583885
F1_score: 0. 968336919766



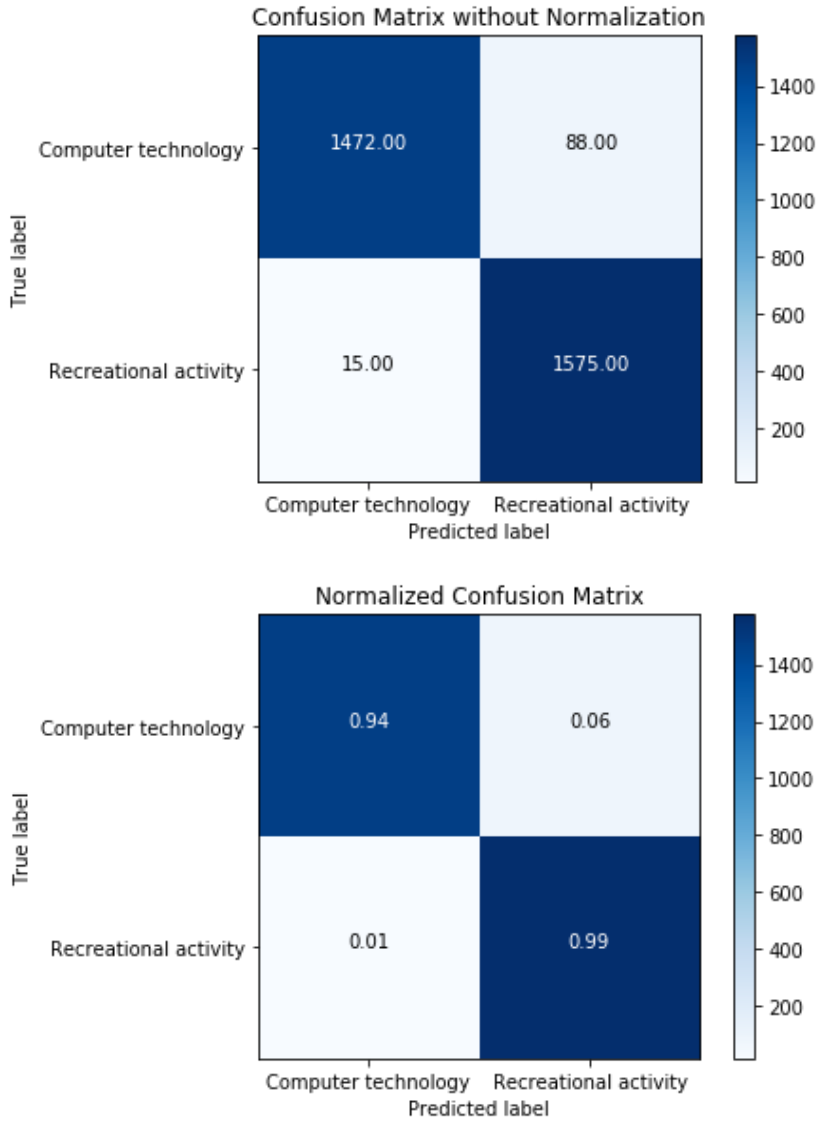


Figure 4: The confusion metric of hard margin SVC with LSI & min_df = 2

————————————————RESULTS of hard margin SVC with LSI & min_df = 5

Accuracy: 0.852063492063

Recall: 0.707547169811

Precision: 0.999111900533

F1_score: 0.828424153166

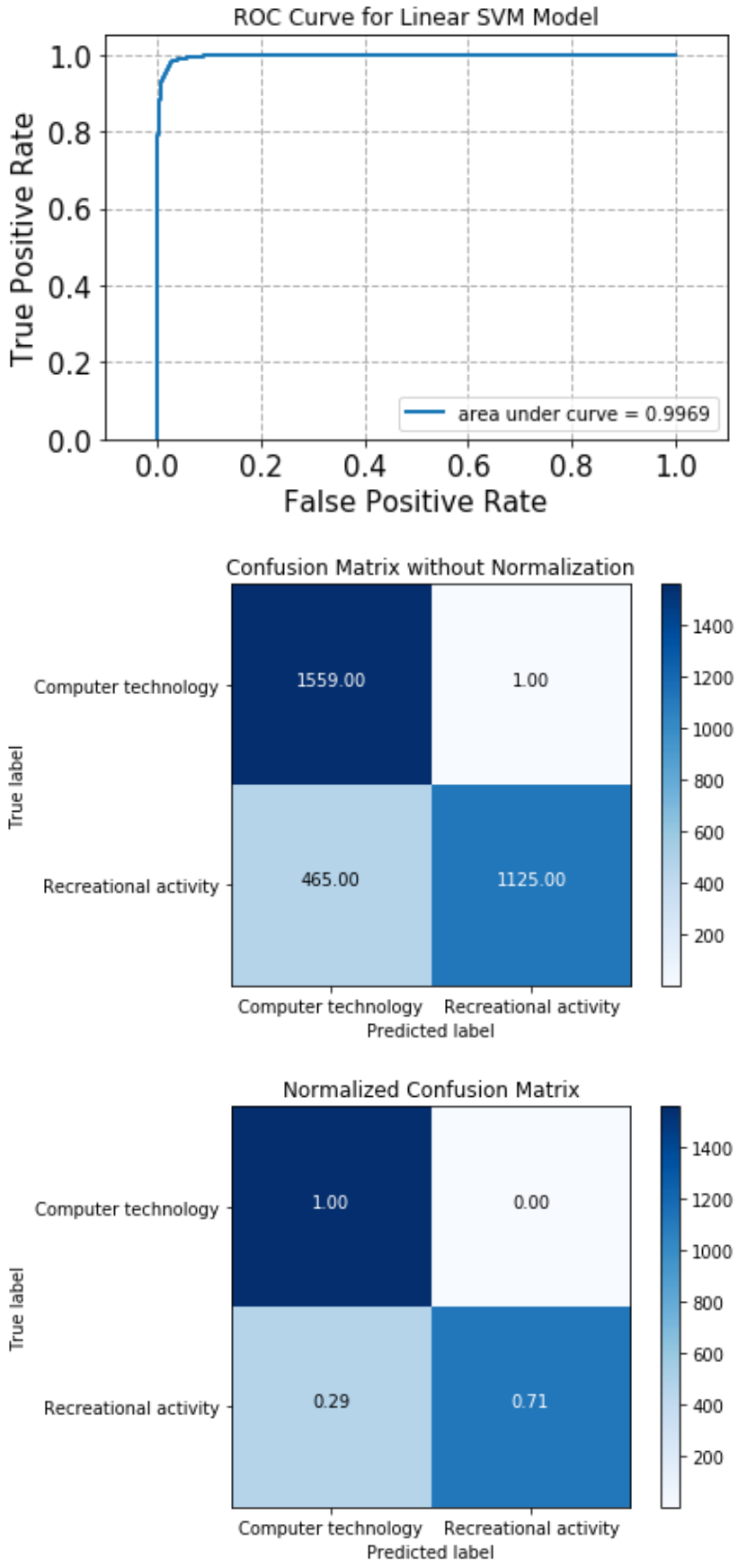


Figure 5: The confusion metric of hard margin SVC with LSI & min_df = 5

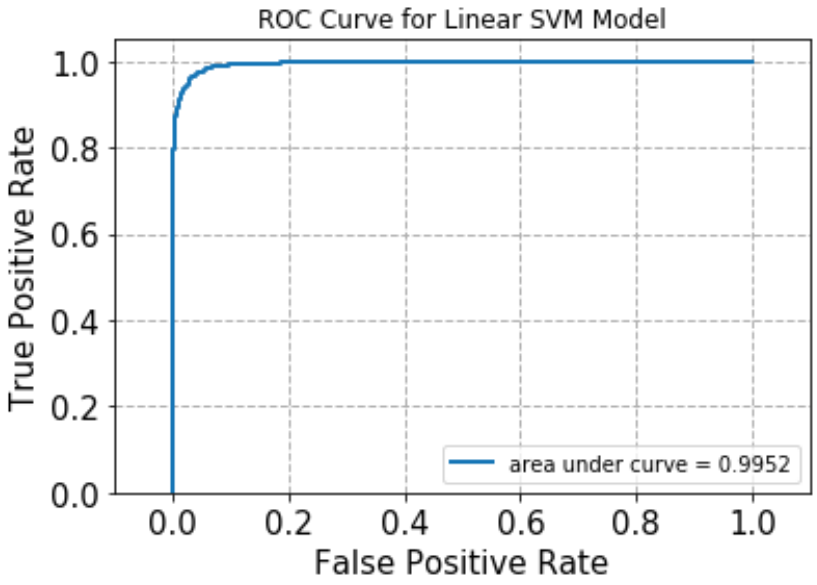
————————————————RESULTS of hard margin SVC with NMF & min_df = 2

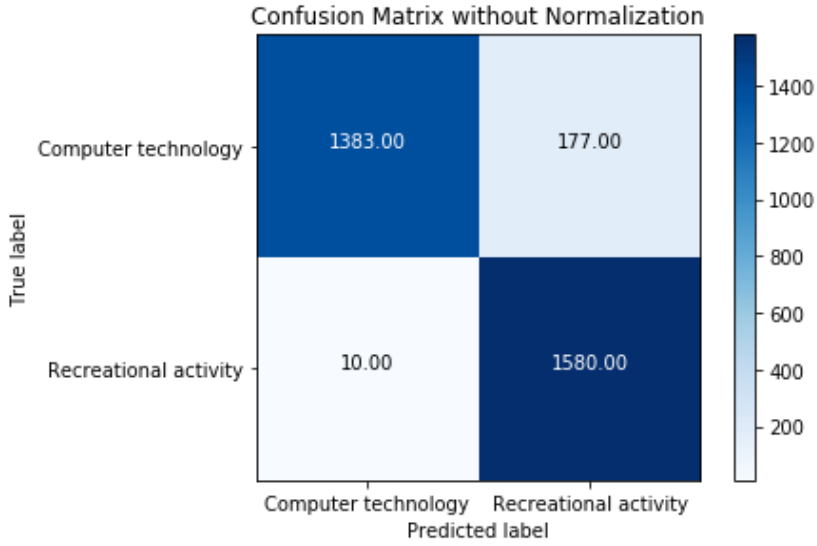
Accuracy: 0. 940634920635

Recall: 0. 993710691824

Precision: 0. 899260102447

F1_score: 0. 94412907081





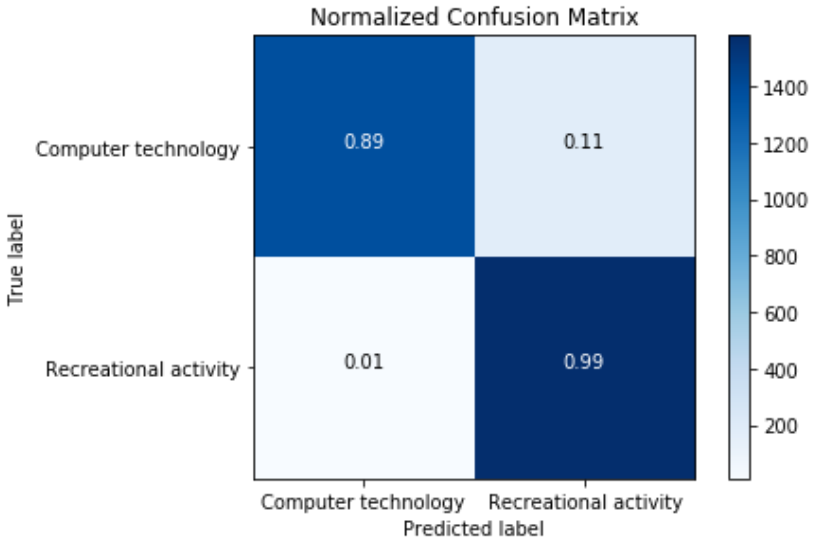


Figure 6: The confusion metric of hard margin SVC with NMF & min_df = 2

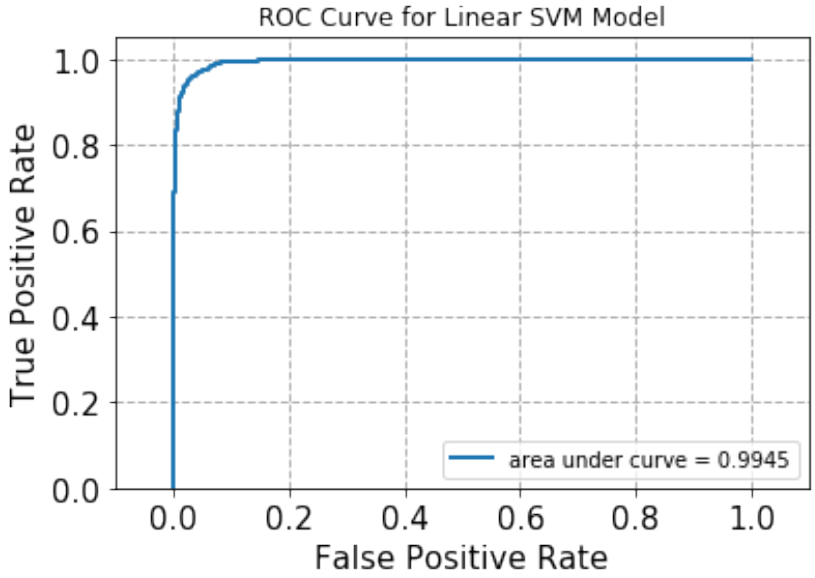
————————————————RESULTS of soft margin SVC with LSI & min_df = 2

Accuracy: 0.938095238095

Recall: 0.994339622642

Precision: 0.894736842105

F1_score: 0.941912421805



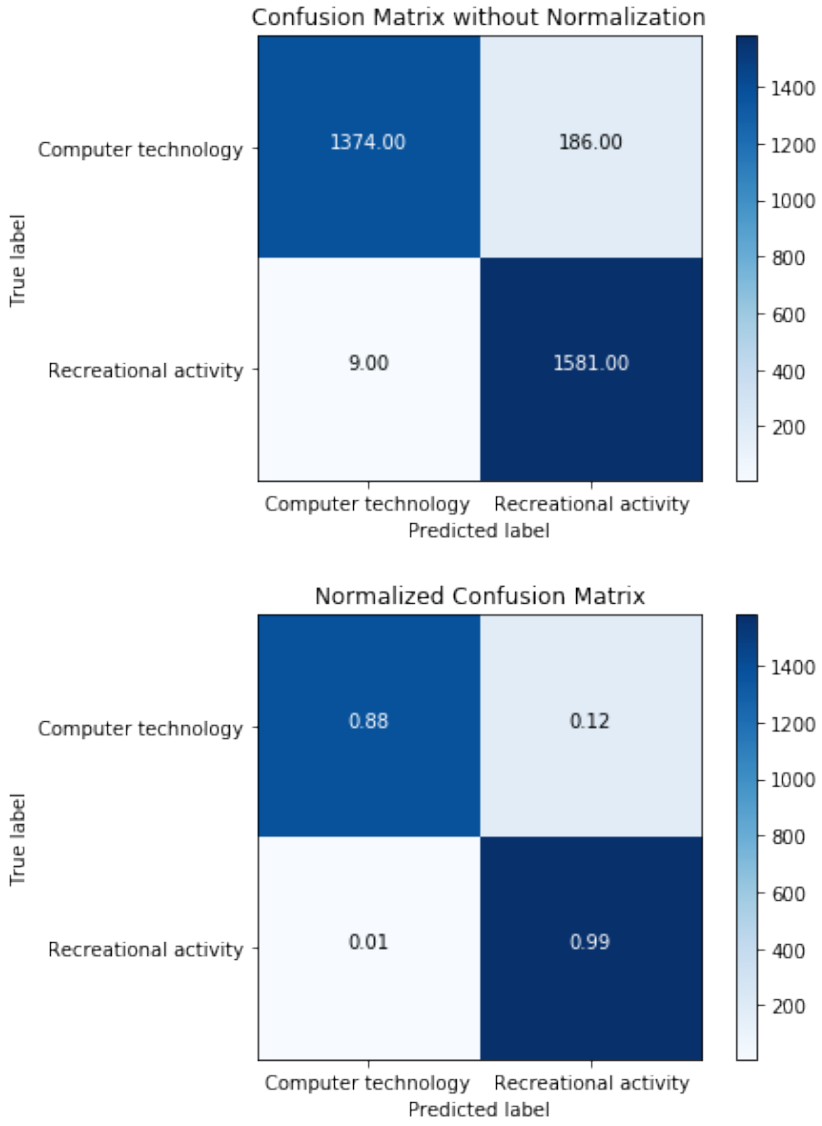


Figure 7: The confusion metric of soft margin SVC with LSI & min_df = 2

————————————RESULTS of soft margin SVC with LSI & min_df = 5

Accuracy: 0.940317460317

Recall: 0.993710691824

Precision: 0.898748577929

F1_score: 0.943847072879

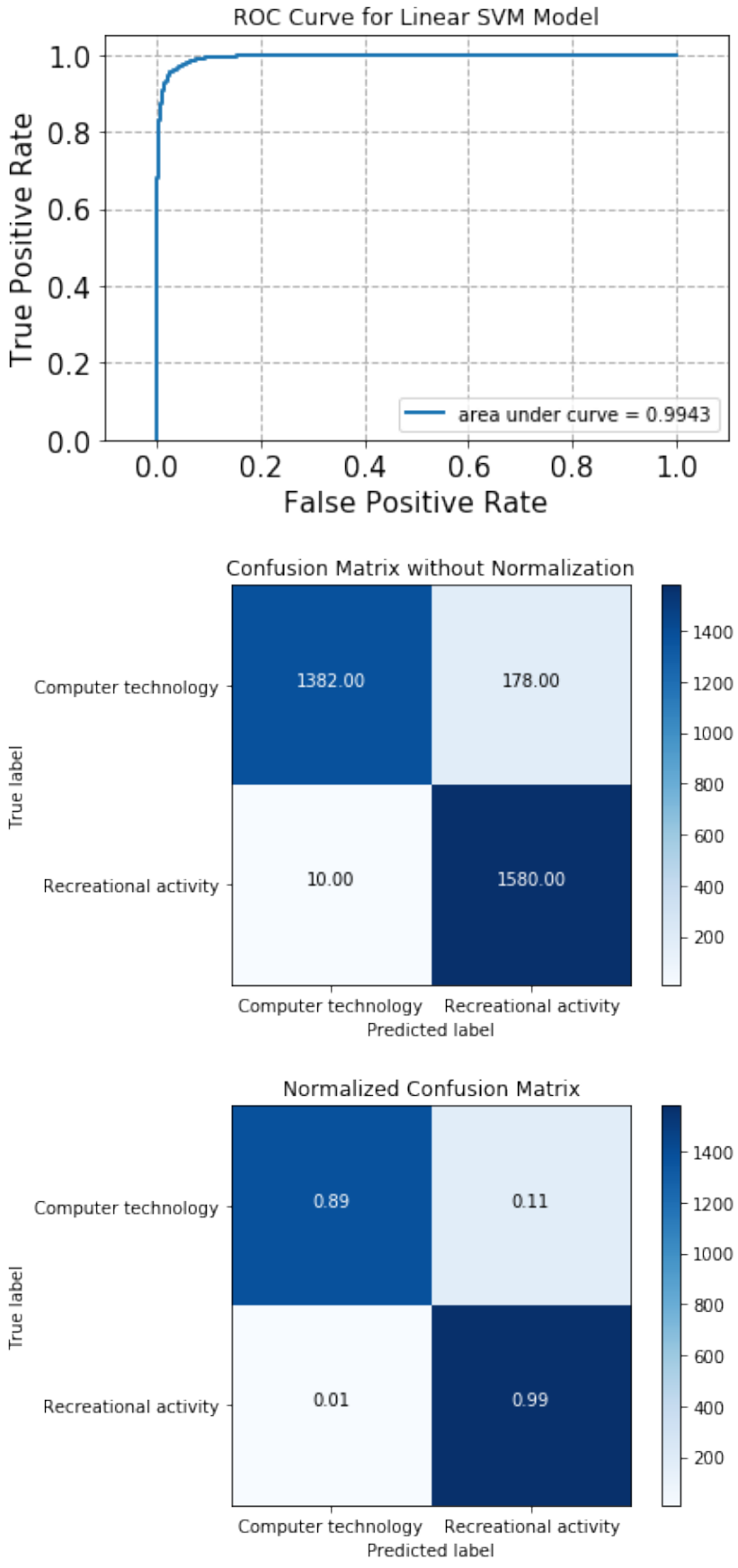


Figure 8: The confusion metric of soft margin SVC with LSI & min_df = 5

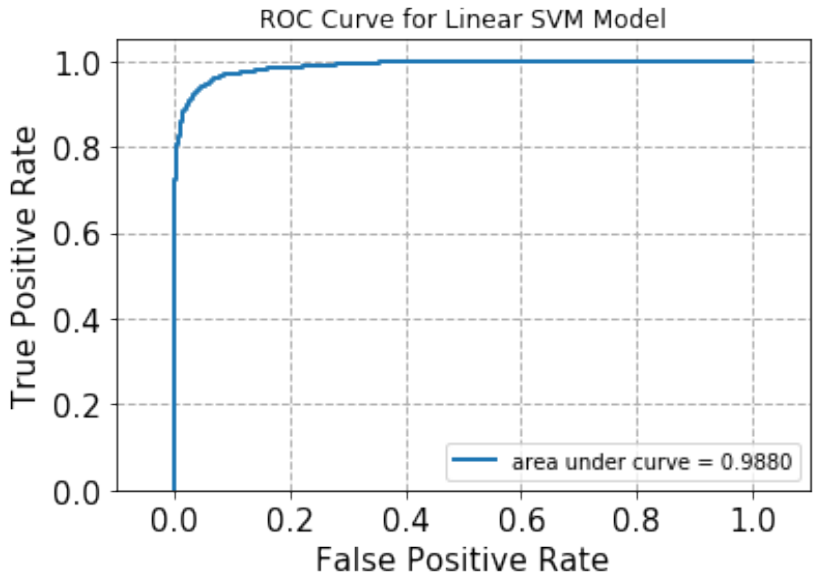
————————————————RESULTS of soft margin SVC with NMF & min_df = 2

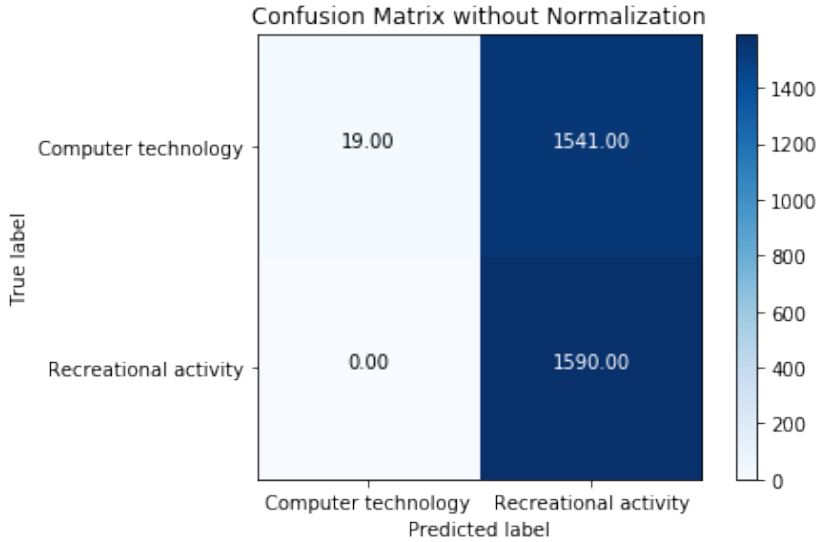
Accuracy: 0.510793650794

Recall: 1.000000000000

Precision: 0.507824976046

F1_score: 0.673586104639





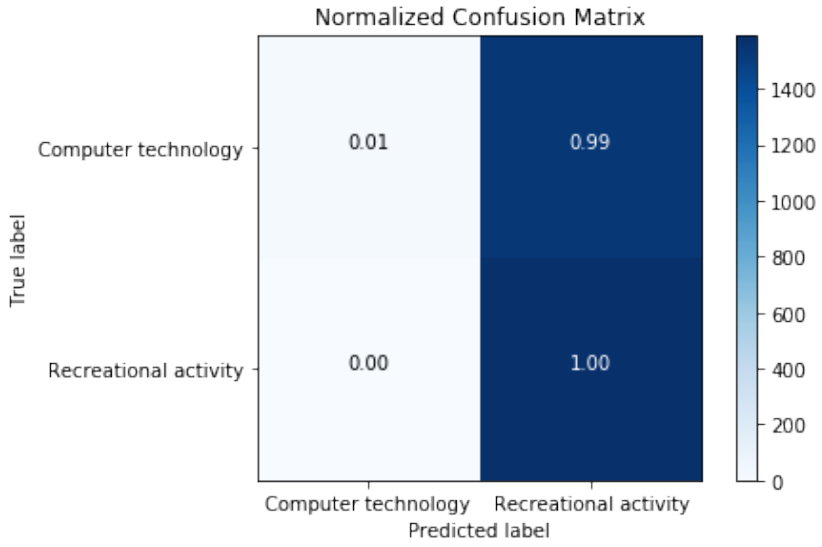


Figure 9: The confusion metric of soft margin SVC with NMF & min_df = 2

Conclusion: Organizing the results gained from the model, we can fill the table with the evaluation coefficients of these different applying situations as shown in Table 2.

	LSI & min_df=2	**LSI & min_df=5**	**NMF & min_df=2**
hard margin SVC	Acc: 0.967 Rec: 0.991 Pre: 0.947 F1s: 0.968	Acc: 0.852 Rec: 0.708 Pre: 0.999 F1s: 0.828	Acc: 0.941 Rec: 0.994 Pre: 0.899 F1s: 0.944
soft margin SVC	Acc: 0.938 Rec: 0.994 Pre: 0.894 F1s: 0.942	Acc: 0.940 Rec: 0.994 Pre: 0.899 F1s: 0.944	Acc: 0.511 Rec: 1.000 Pre: 0.508 F1s: 0.674

Table 2: The results of SVC model in different situations

From the organized table results, it is obvious that SVM classification is a pretty good model. And to be more specific, except the soft margin SVC with NMF and min_df of 2, in the other situations all the accuracies, the recalls, the precisions and F1_scores of linear SVC can reach more than 0.8 which is a relatively high value for classification model.

For more details from comparing the evaluation coefficients in different situations, we can draw conclusions on how the parameters affect the results. For hard margin SVC, using LSI or

NMF for dimension reduction and whether min_df is high or low are comparatively irrelevant since only very slight decrease of accuracy and F1_socre occurs. Overall values of the soft margin SVC are lower than those of hard margin SVC. However, NMF brings very dramatic influence to the results. The recall increase to 1 while accuracy and the precision go down to nearly 0.5, probably because the model makes too much positive predictions. Thus, we could say that, NMF is never an ideal dimension reduction method for soft margin SVC.

Problem F

Requirement: Using a 5-fold cross-validation, find the best value of the parameter γ in the range $10^{\mathrm{H}} - 3 \leq k \leq 3, k \in Z\}$. Report the confusion matrix and calculate the accuracy, recall and precision of the classifier.

In Problem F, we still use Linear SVM as the classifier, which is similar to what we do in Problem E. However, we are required to apply 5-fold cross validation to tune the tradeoff parameter C, which represents γ in the optimization. By applying 5-fold cross validation, we could split the training data into 5 parts. Then we train the model using the rest 4 of them with a given hyper parameter C and validate the performance with the part of training data. After choosing every fold of data as validation data we could get five scores. Averaging the scores, we could evaluate the performance of models with different hyper parameters.

Result: After using the iteration, we find the best gamma value as 100 of all the situations in this problem. And the confusion matrix and the evaluation coefficients of each situation are attached behind, as shown in Figure 10-13 and Table 3.

————————————————RESULTS of linear SVC with LSI & min_df = 2

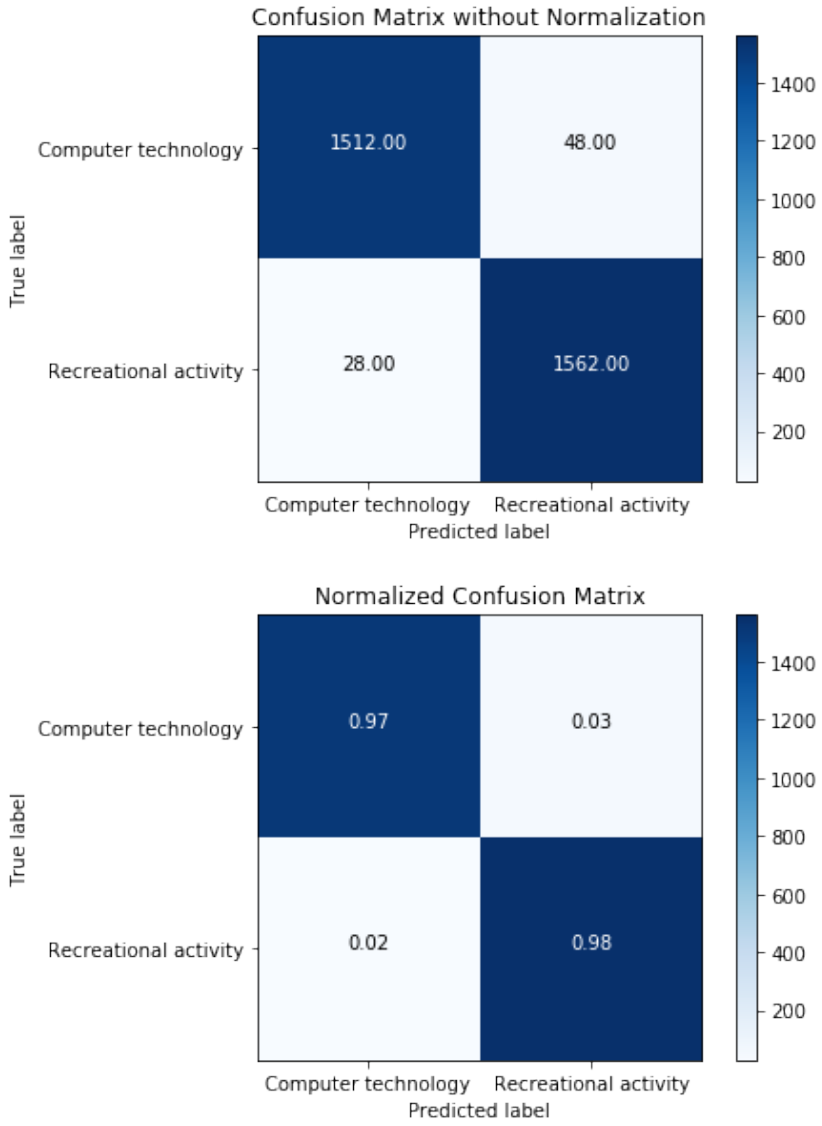


Figure 10: The confusion metric of linear SVC with LSI & min_df = 2

————————————————RESULTS of linear SVC with LSI & min_df = 5

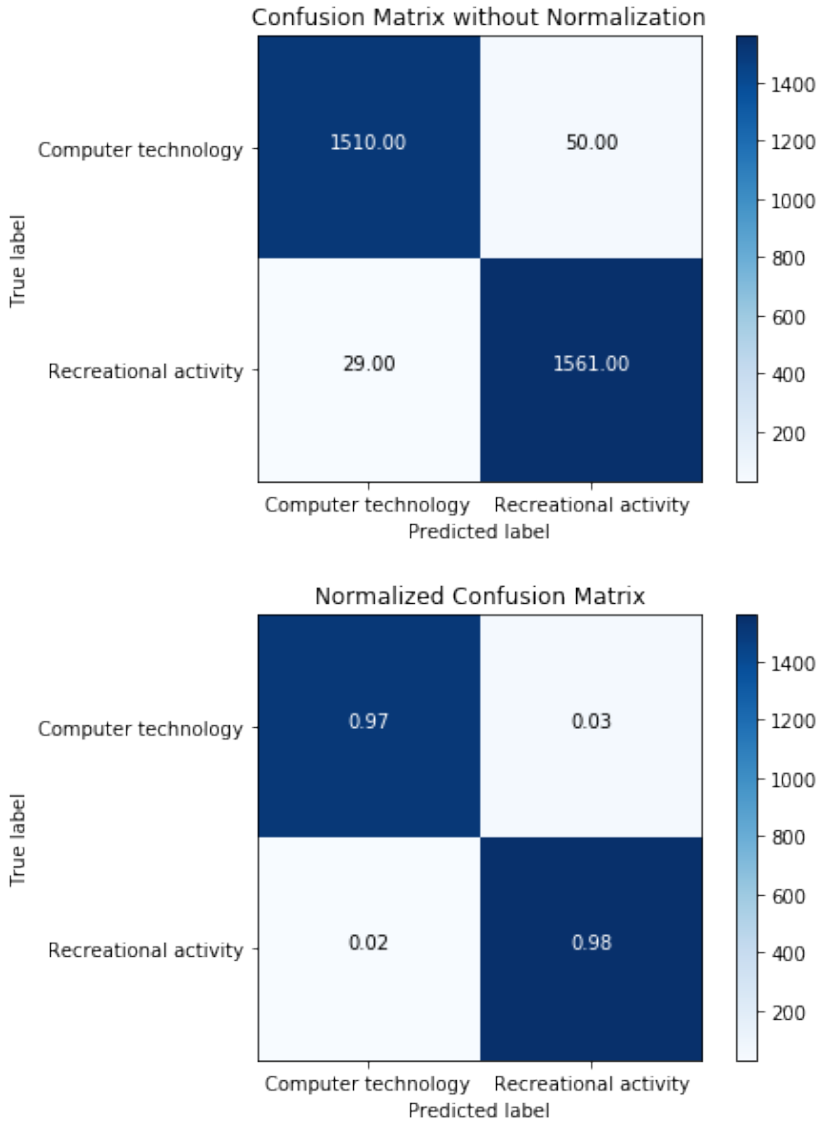


Figure 11: The confusion metric of soft margin SVC with LSI & min_df = 5

————————————RESULTS of linear SVC with NMF & min_df = 2

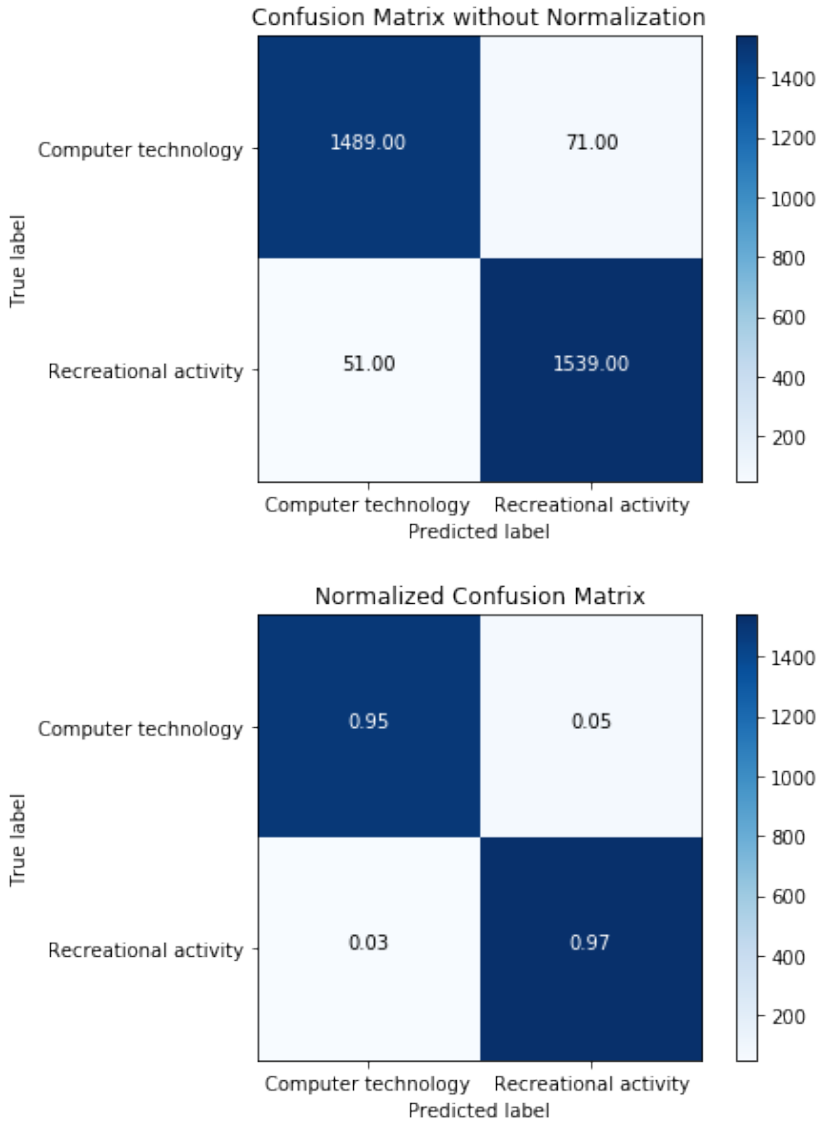


Figure 12: The confusion metric of linear SVC with NMF & min_df = 2

	LSI & min_df=2	**LSI & min_df=5**	**NMF & min_df=2**
Best gamma value	100	100	100
Evaluation coefficients with the best regulation parameter	Acc: 0.976 Rec: 0.982 Pre: 0.970 F1s: 0.976	Acc: 0.975 Rec: 0.982 Pre: 0.969 F1s: 0.975	Acc: 0.961 Rec: 0.968 Pre: 0.956 F1s: 0.962

Table 3: The results of best gamma value of SVC model in different situations

Conclusion: Table 3 shows that the different models with different methods of dimension reduction (LSI or NMF) as well as the values of min_df all work best when C = 100. The possible reason is that the model may suffer overfitting when C is too large while a small C might lead to under-fitting.

And compare the results of the evaluation coefficients when using the best regulation parameter but with different methods of dimension reduction and the values of min_df, we could find out that the three classifiers all work excellent, whose F1_scores are all over 0.96. To be more specific, the linear SVC with LSI & min_df = 2 works slightly better than the rest two classifiers.

Problem G

Requirement: Next, we use naïve Bayes algorithm for the same classification task. The algorithm estimates the maximum likelihood probability of a class given a document with feature set x, using Bayes rule, based upon the assumption that given the class, the features are statistically independent. Train a multinomial 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.

The steps in Problem G are almost the same as those in Problem E. However, there are still several differences. The first difference is, of course, the classifier. We are requested to use naïve Bayes algorithm as the classifier. And in order to learn the effects of different classifiers, we also applied GaussianNB in addition to MultinomialNB. However, MultinomialNB classifier could not classify data after applying dimension reduction to the data since there would be negative values while the MultinomialNB restricts that the input must be non-negative. Recording to some references and asking for the TA's suggestion, we choose to skip the step of dimension reduction when using MultinomialNB. Then, we just follow the normal steps to get the ROC curve as well as the confusion matrix and calculate the accuracy, recall and precision of the classifiers and compare the results with different coefficients.

Results: The results are divided into MultinomialNB and GaussianNB

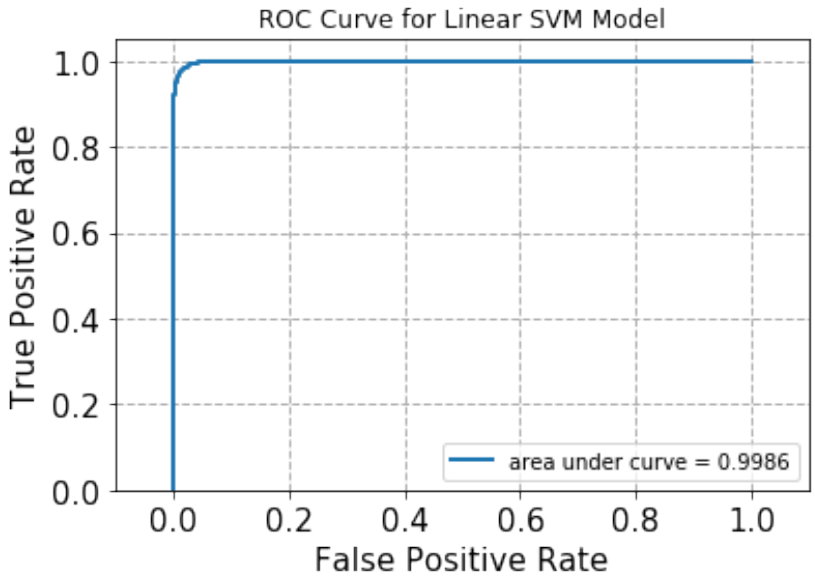
————————————————RESULTS of MultinomialNB with min_df = 2

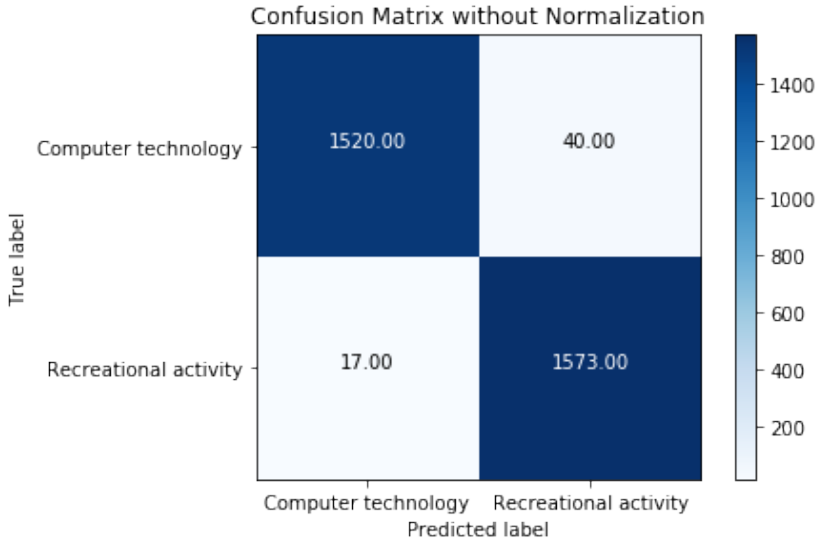
Accuracy: 0.981904761905

Recall: 0.989308176101

Precision: 0.975201487911

F1_score: 0.982204183578





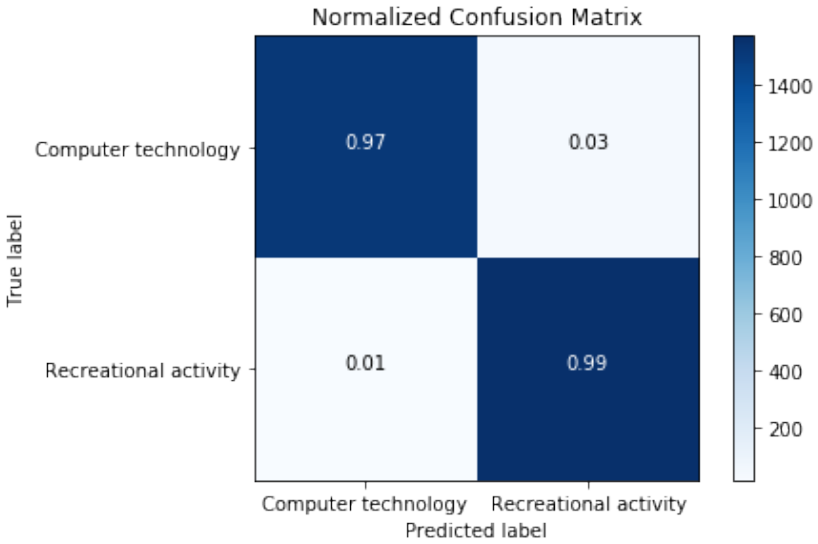


Figure 13: The confusion metric of MultinomialNB with min_df = 2

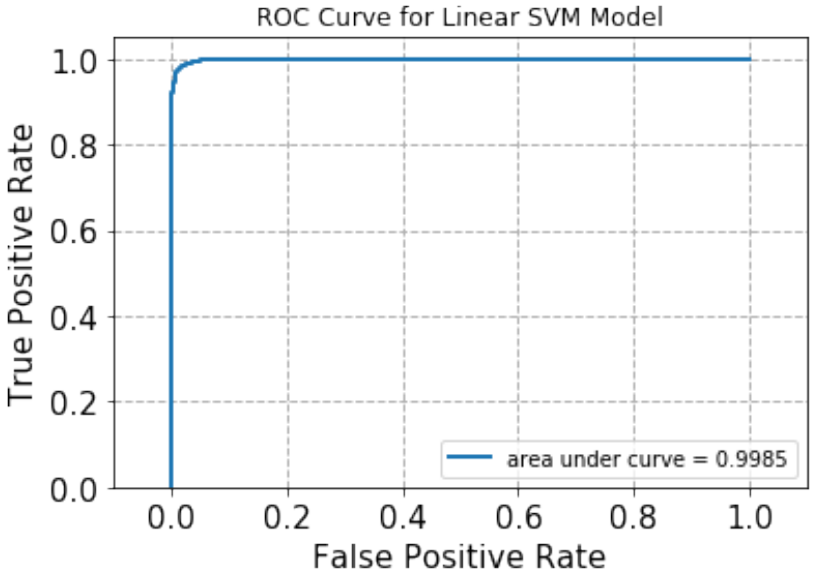
————————————————RESULTS of MultinomialNB with min_df = 5

Accuracy: 0.980952380952

Recall: 0.981132075472

Precision: 0.981132075472

F1_score: 0.981132075472



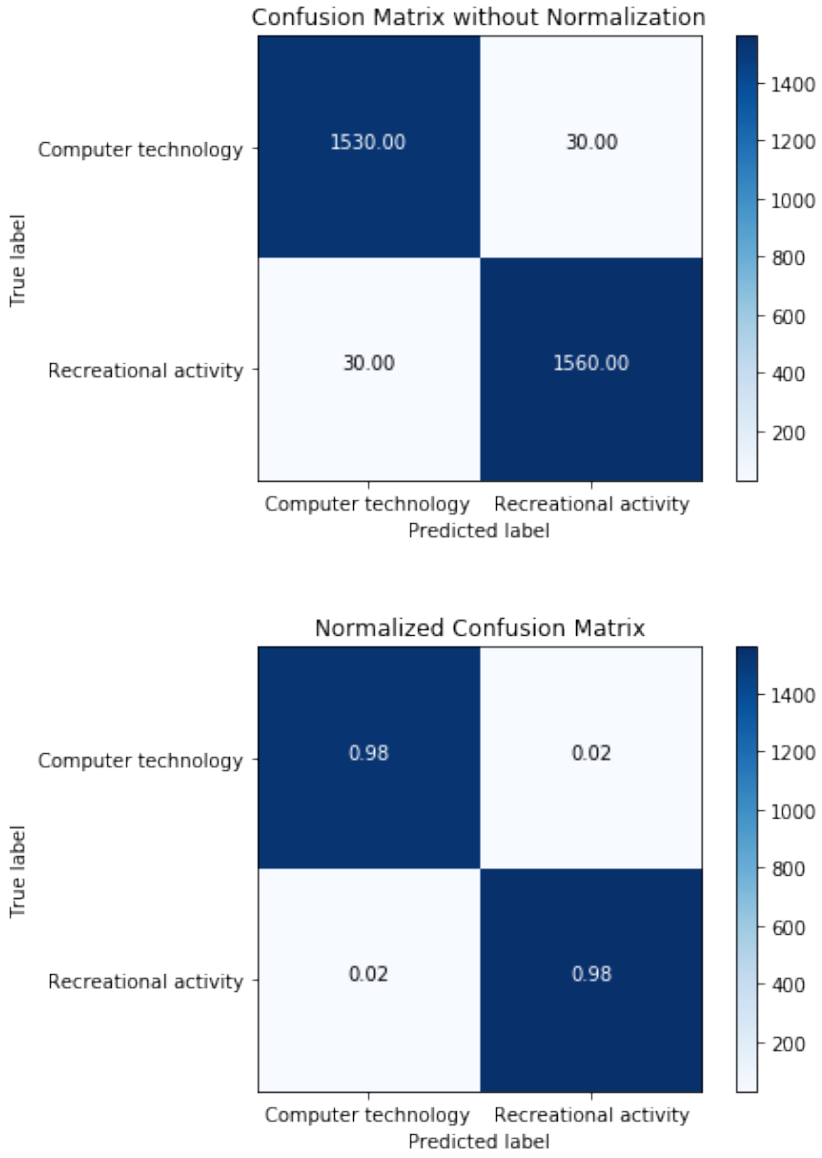


Figure 14: The confusion metric of MultinomialNB with min_df = 5

————————————————RESULTS of GaussianNB with LSI & min_df = 2

Accuracy: 0.905714285714

Recall: 0.945911949686

Precision: 0.876967930029

F1_score: 0.910136157337

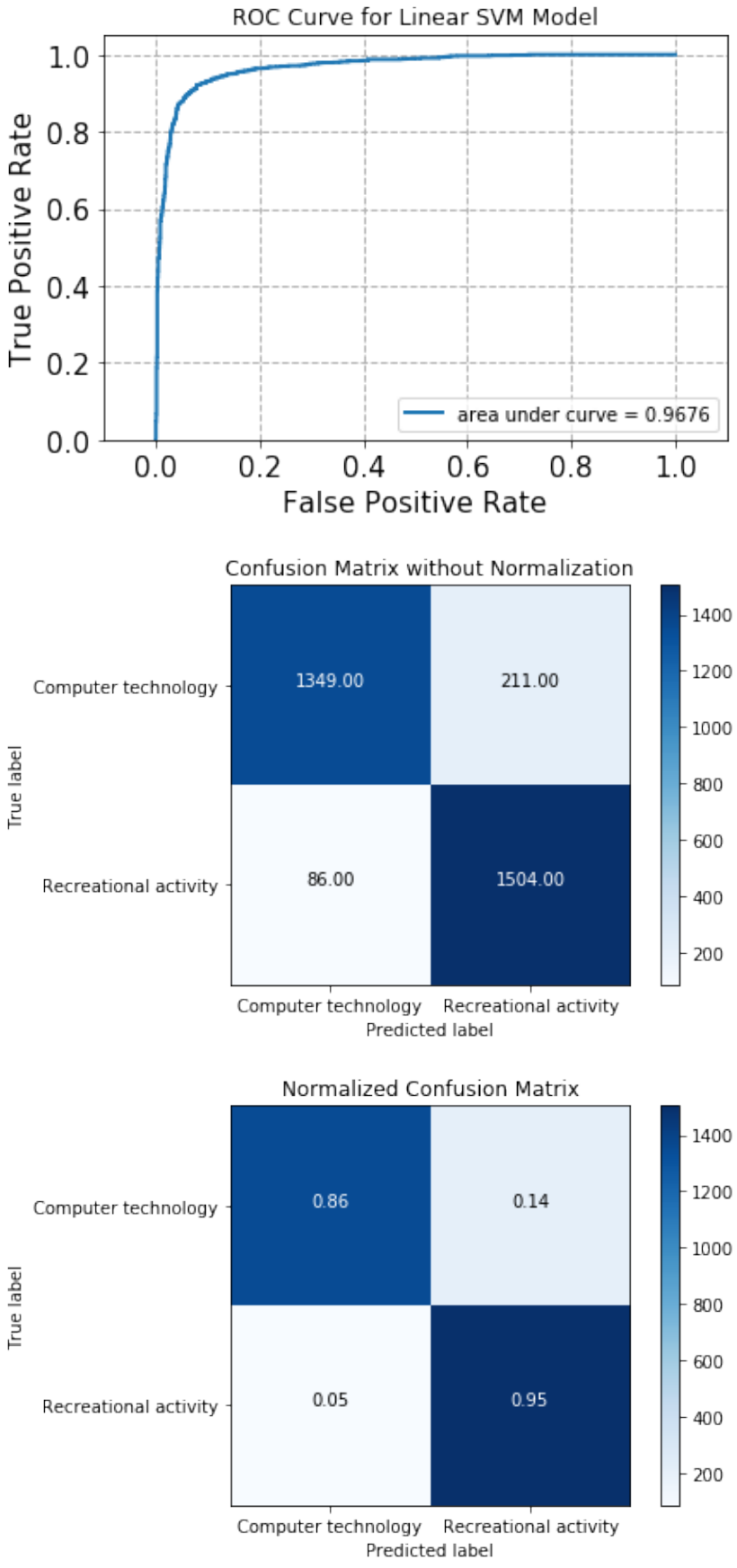


Figure 15: The confusion metric of GaussianNB with LSI & min_df = 2

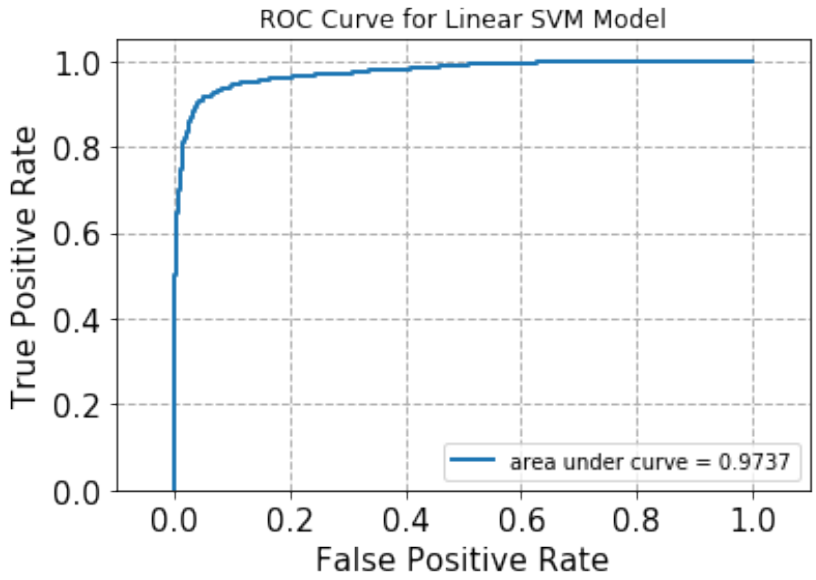
————————————RESULTS of GaussianNB with LSI & min_df = 5

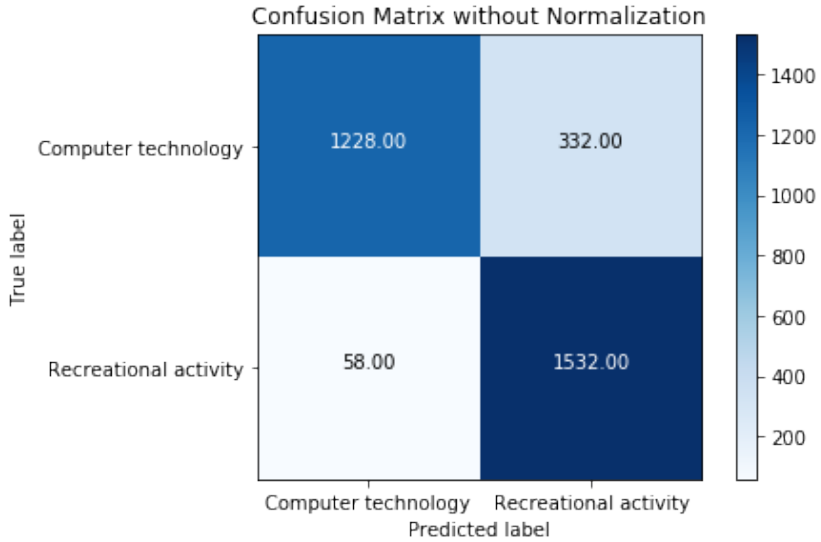
Accuracy: 0.87619047619

Recall: 0.963522012579

Precision: 0.821888412017

F1_score: 0.887087434858





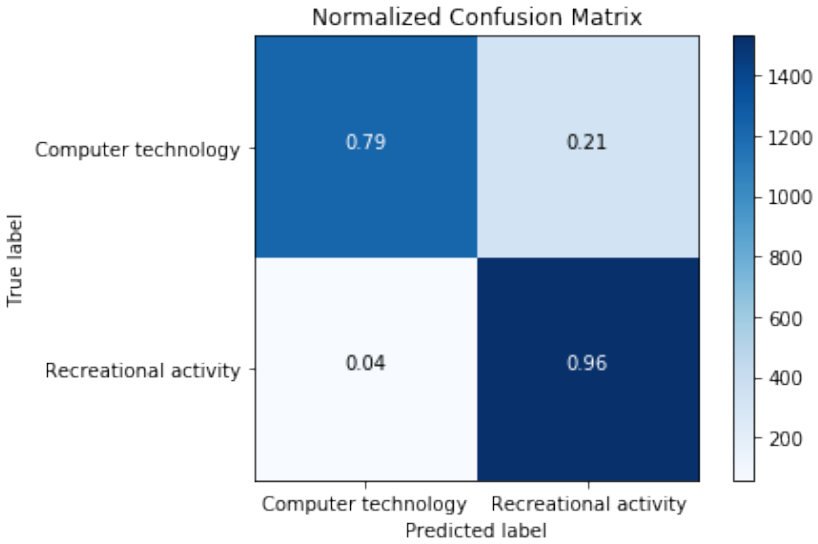


Figure 16: The confusion metric of GaussianNB with LSI & min_df = 5

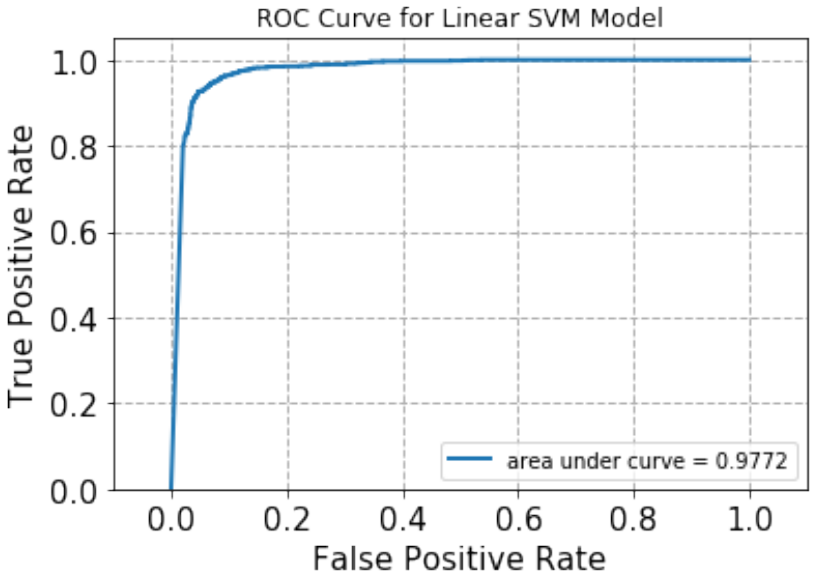
————————————————RESULTS of GaussianNB with NMF & min_df = 2

Accuracy: 0.934920634921

Recall: 0.960377358491

Precision: 0.914919113241

F1_score: 0.937097269101



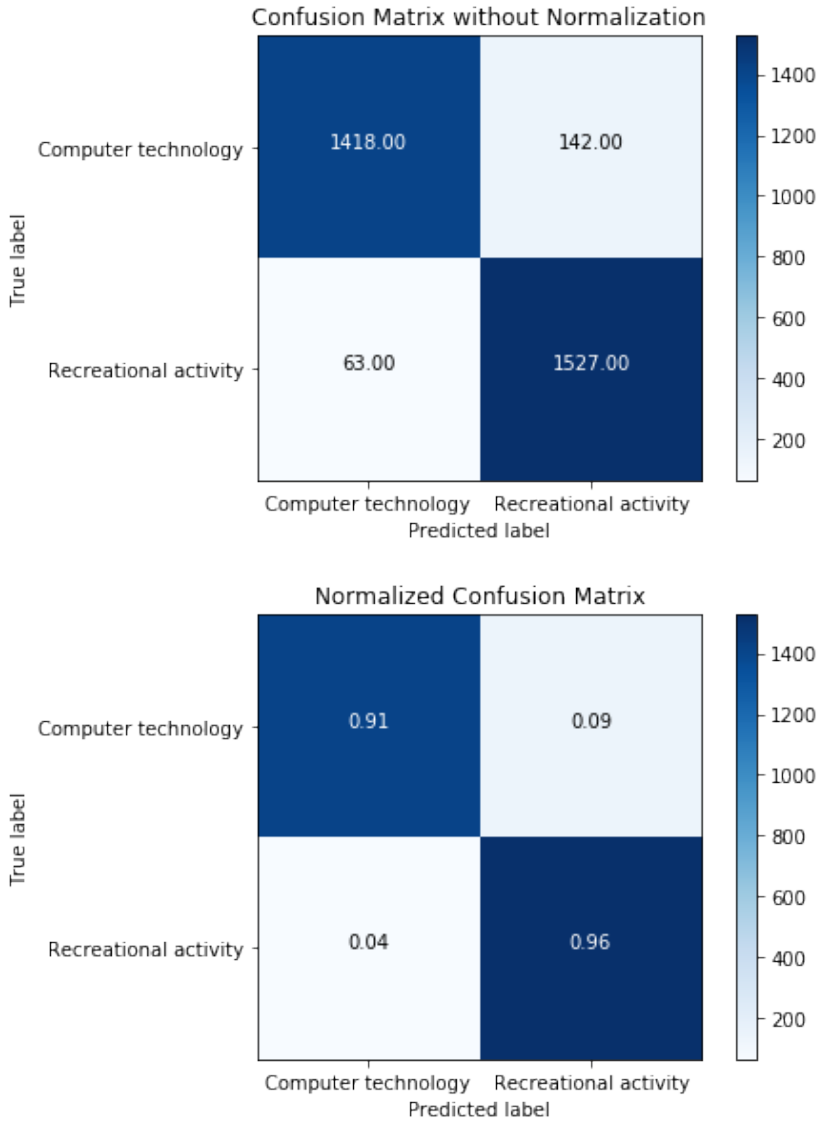


Figure 17: The confusion metric of GaussianNB with NMF & min_df = 2

Conclusion: Organizing the results gained from the model, we can fill the table with the evaluation coefficients of these different applying situations as shown in Table 4.

	At min_df of 2	**At min_df of 5**
MultinomialNB	Acc: 0.982 Rec: 0.989 Pre: 0.975 F1s: 0.982	Acc: 0.981 Rec: 0.981 Pre: 0.981 F1s: 0.981

	Using LSI	Using NMF	Using LSI
GaussianNB	Acc: 0.906 Rec: 0.946 Pre: 0.877 F1s: 0.910	Acc: 0.935 Rec: 0.960 Pre: 0.915 F1s: 0.937	Acc: 0.876 Rec: 0.964 Pre: 0.822 F1s: 0.887

Table 4: The results of naïve Bayes model in different situations

From the organized table results, it is obvious that MultinomialNB classification model has better performance than the GaussianNB classification model, and the highest value is shown in MultinomialNB at min_df of 2. However, if we focus on the GaussianNB classification model, we can't say it's a bad one since every evaluation coefficient is near to 0.9.

After we take a closer look to the performances of these two classification models in different situation, we can find that the parameter min_df is relatively irrelevant to MultinomialNB while comparatively the Gaussian can be affected more by this parameter. The dimension reduction isn't involved in the MultinomialNB model but only plays a role in GaussianNB model. Applying GaussianNB to the data processed by LSI or NMF, we can tell from the results that, NMF is a more ideal method to process the data for GaussianNB than the LSI is.

Problem H

Requirement: 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.

In Problem H, we just repeat the same task but with the logistic regression classifier. And we still use the pipeline to train and predict then get the ROC curve and calculate the accuracy, recall and precision of the classifier so as to compare the logistic regression classifier with other classifiers.

Results: The results are listed as follows.

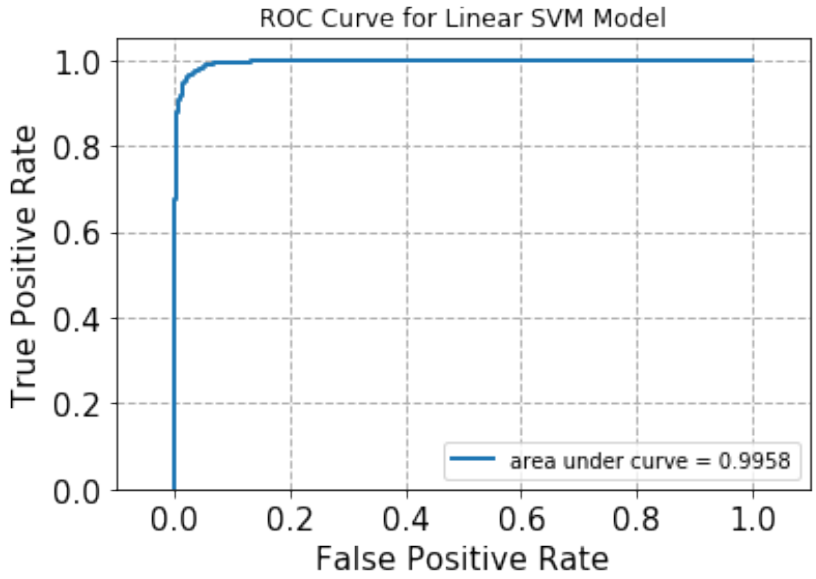
————————————————RESULTS of logistic regression classifier with LSI & min_df = 2

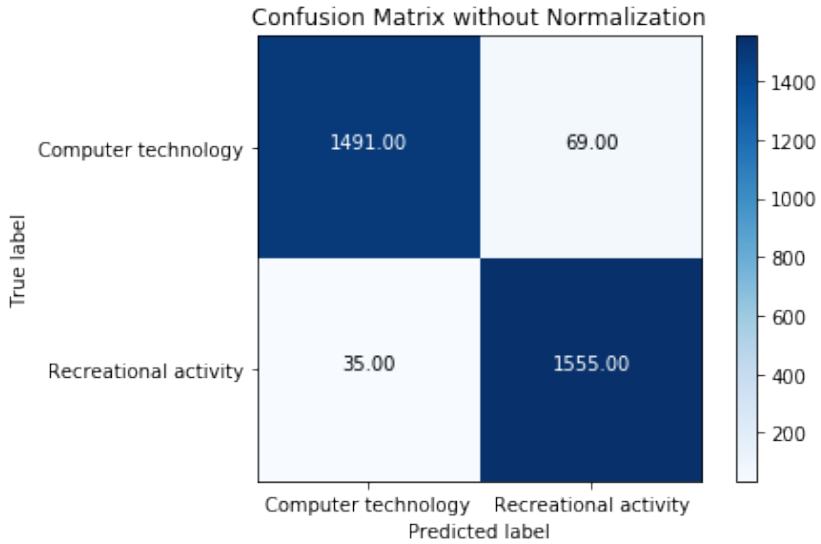
Accuracy: 0.966984126984

Recall: 0.977987421384

Precision: 0.957512315271

F1_score: 0.967641568139





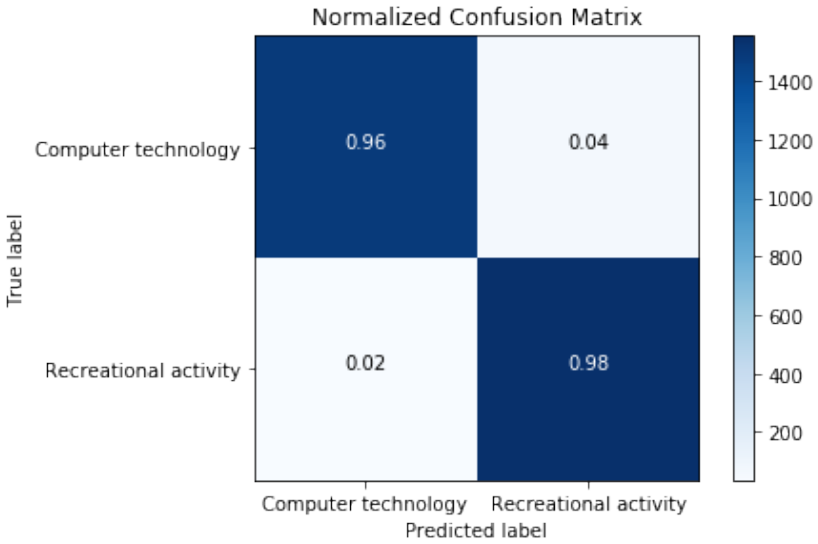


Figure 18: The confusion metric of logistic regression classifier with LSI & min_df = 2

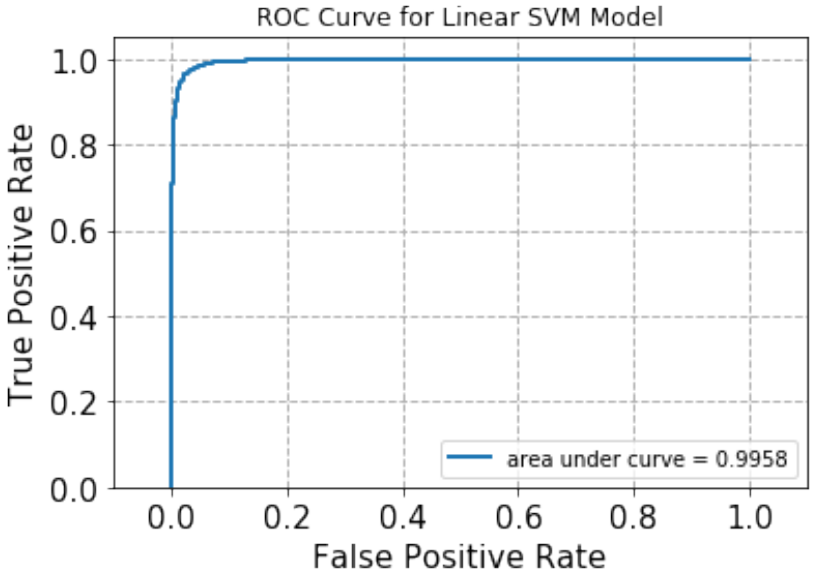
————————————————RESULTS of logistic regression classifier with LSI & min_df = 5

Accuracy: 0.968571428571

Recall: 0.979245283019

Precision: 0.959334565619

F1_score: 0.96918767507



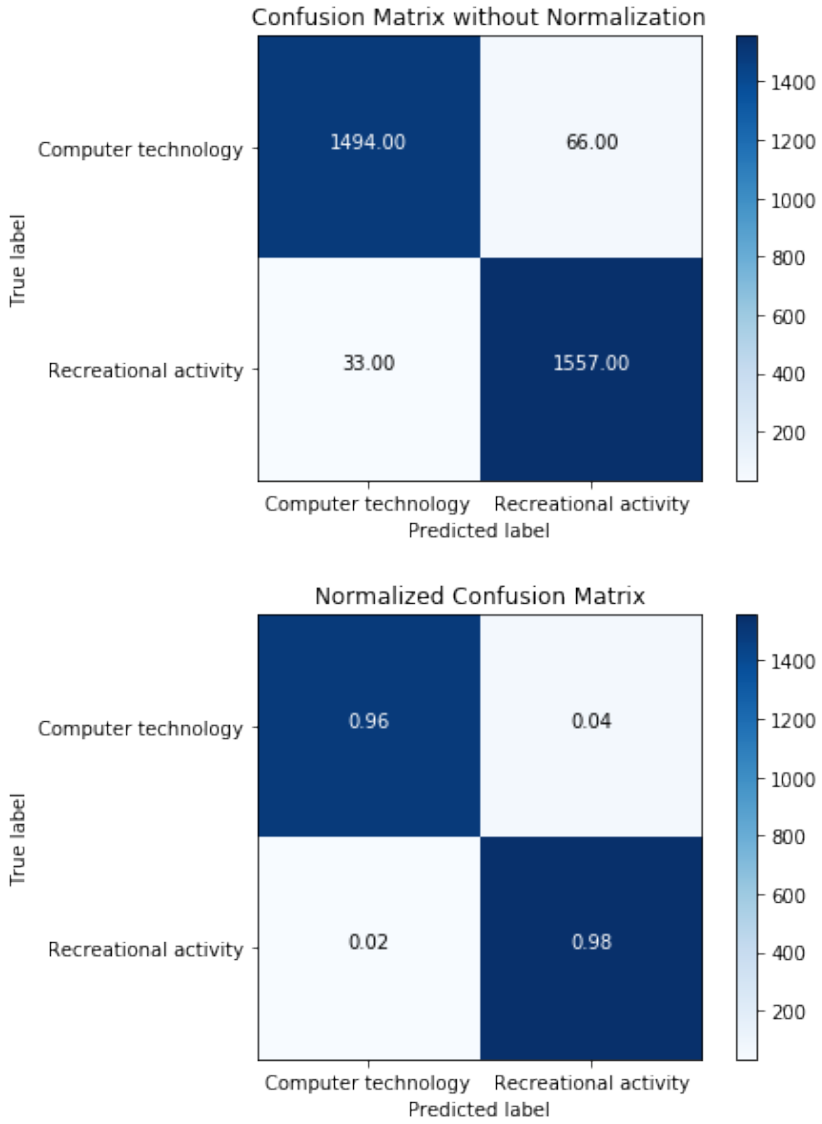


Figure 19: The confusion metric of logistic regression classifier with LSI & min_df = 5

————————————RESULTS of logistic regression classifier with NMF & min_df = 2

Accuracy: 0.952698412698

Recall: 0.952830188679

Precision: 0.953429830082

F1_score: 0.953129915068

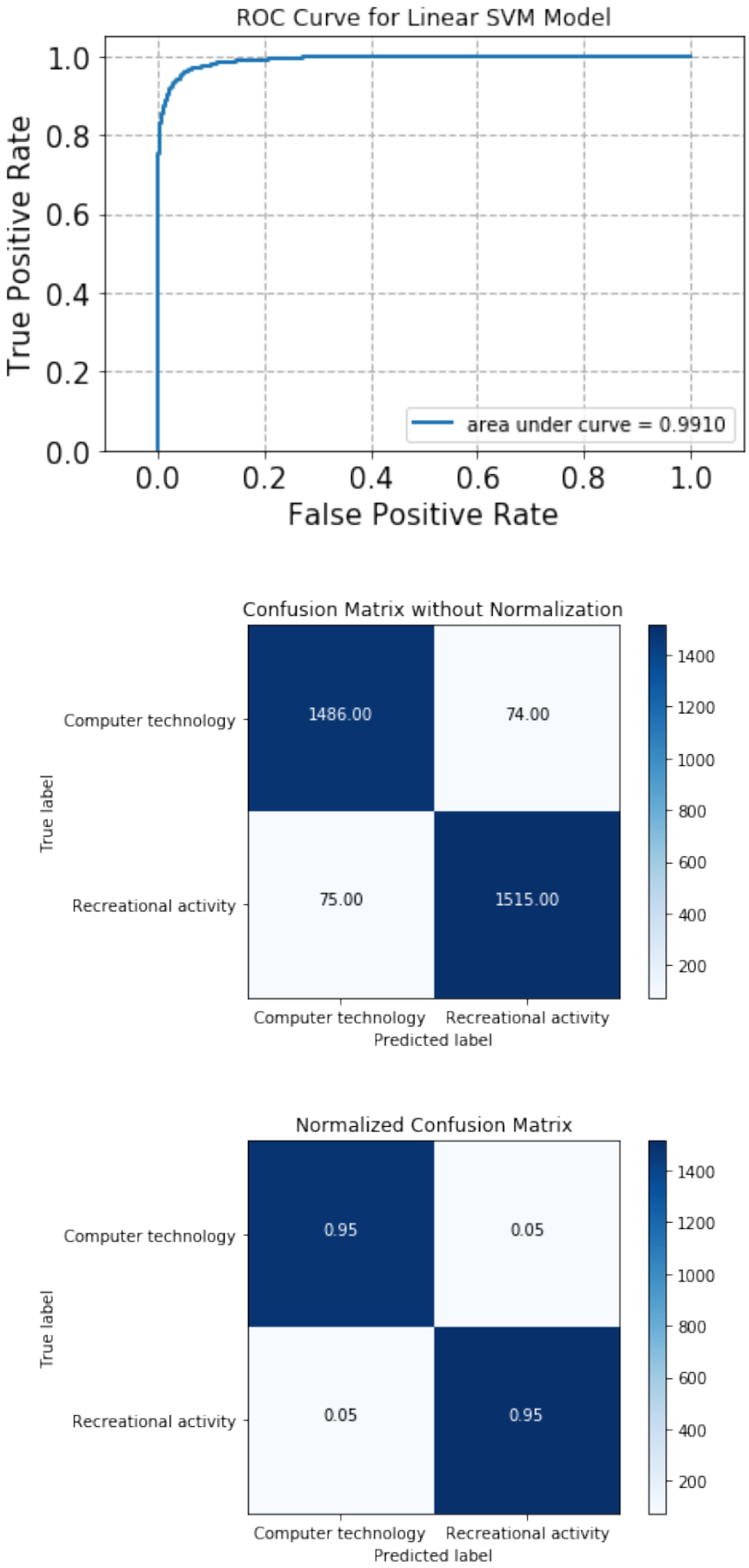


Figure 20: The confusion metric of logistic regression classifier with NMF & min_df = 2

Conclusion: The table filled with the organized results is listed as Table 5.

	LSI & min_df=2	LSI & min_df=5	NMF & min_df=2
logistic regression	Acc: 0.967 Rec: 0.978 Pre: 0.958 F1s: 0.968	Acc: 0.969 Rec: 0.979 Pre: 0.959 F1s: 0.969	Acc: 0.953 Rec: 0.953 Pre: 0.953 F1s: 0.953

Table 5: The results of logistic regression classifier in different situations

From the organized table results, it is obvious that logistic regression classifier is also a pretty good model. No matter reducing the dimension through LSI or NMF, the evaluation coefficients of these classifications all show to be excellent even with different min_df. And the F1_scores of these three classifiers are all more than 0.95, which is a relatively high value of classification model.

In addition, the difference between the three classifiers is quite slight. And the logistic regression with LSI & min_df = 2 is relatively the best.

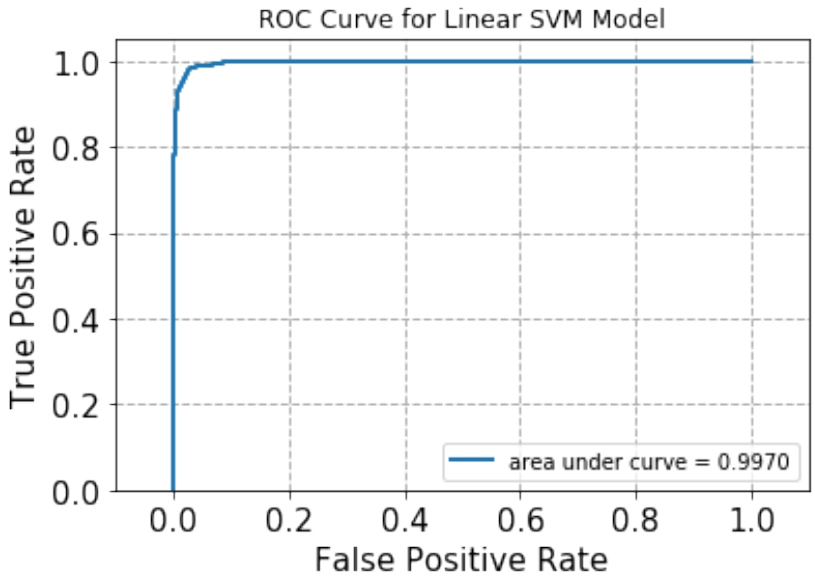
Problem I

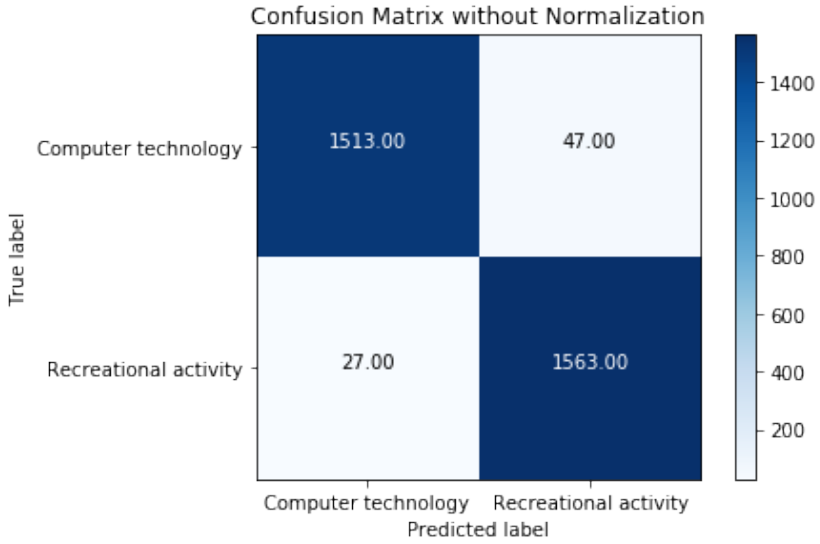
Requirement: Now, repeat part (h) by adding a regularization term to the optimization objective. Try both l_1 and l_2 norm regularizations and sweep through different regularization coefficients, ranging from very small ones to large ones. How does the regularization parameter affect the test error? How are the coefficients of the fitted hyperplane affected? Why might one be interested in each type of regularization?

Since the logistic regression has two penalties 'l1' and 'l2', it is necessary for us to find out how different the results could be when applying different penalties. In addition, we could also set up the different inverse of regularization strength to find out the effect of the coefficient 'C' on the result. Then we could find out the best regularization coefficient and show the ROC curve, the confusion matrix, the accuracy, recall and precision of the classifier with the best regularization coefficient of both 'l1' and 'l2'.

Result: The results are listed as follows.

————————————RESULTS of l_1 normal logistic regression with LSI & min_df = 2





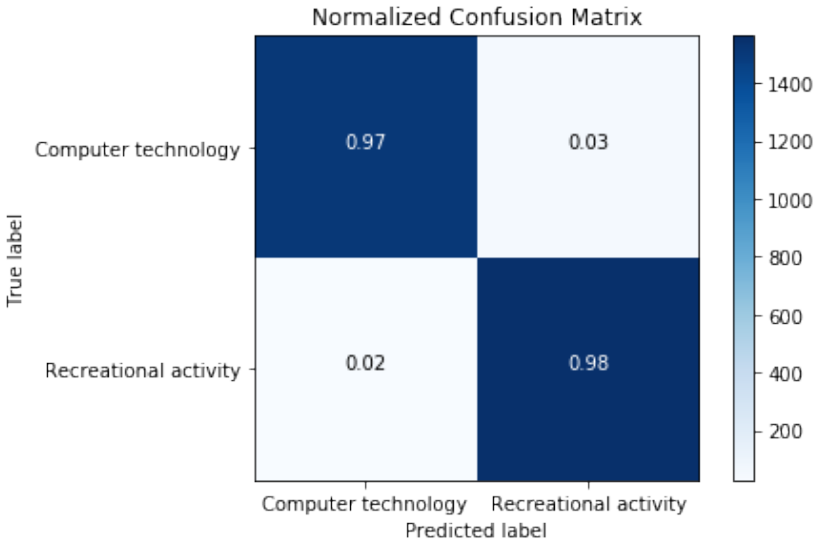
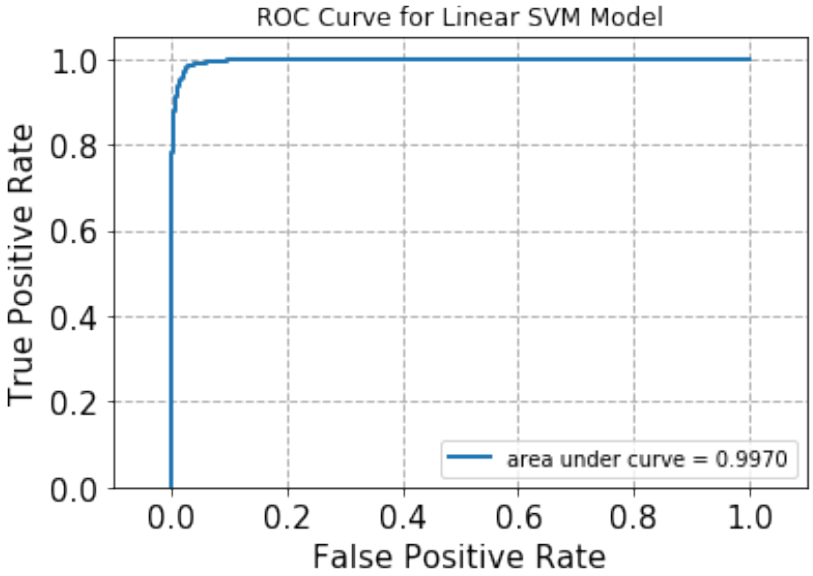


Figure 21: The confusion metric of l1 normal logistic regression with LSI & min_df = 2

————————————————RESULTS of l_1 normal logistic regression with LSI & min_df = 2



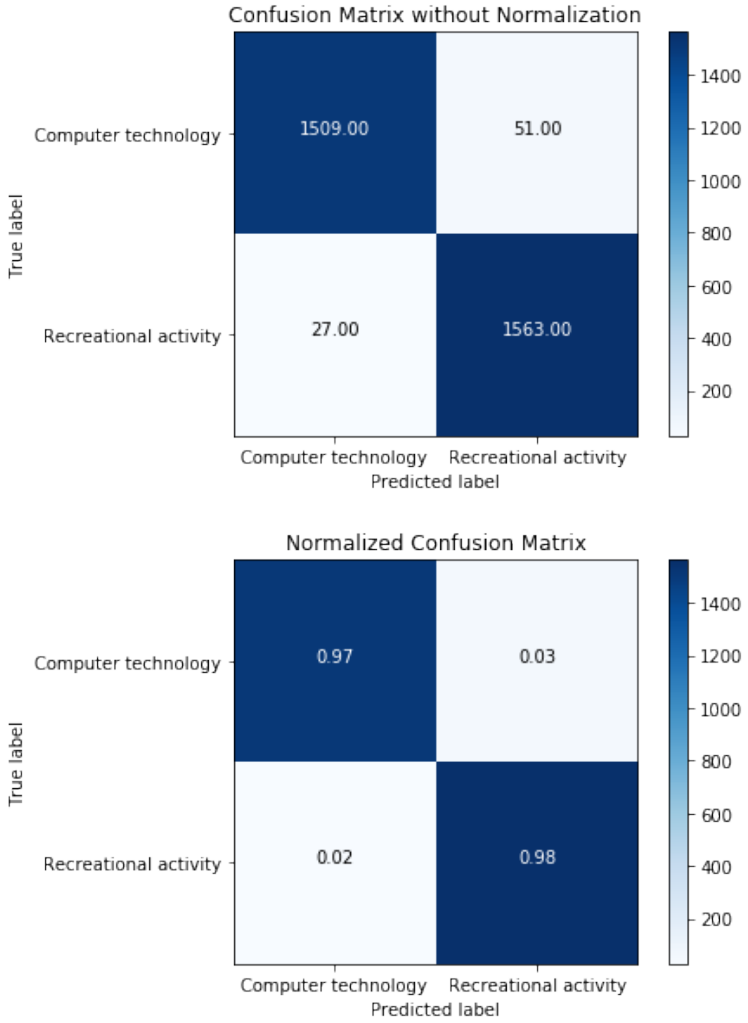
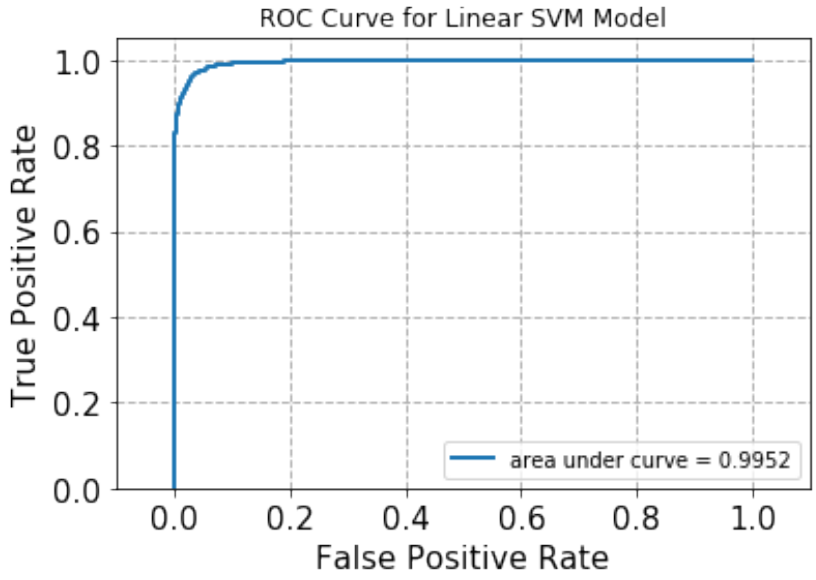


Figure 22: The confusion metric of l1 normal logistic regression with LSI & min_df = 5

————————————————RESULTS of l_1 normal logistic regression with NMF & min_df = 5



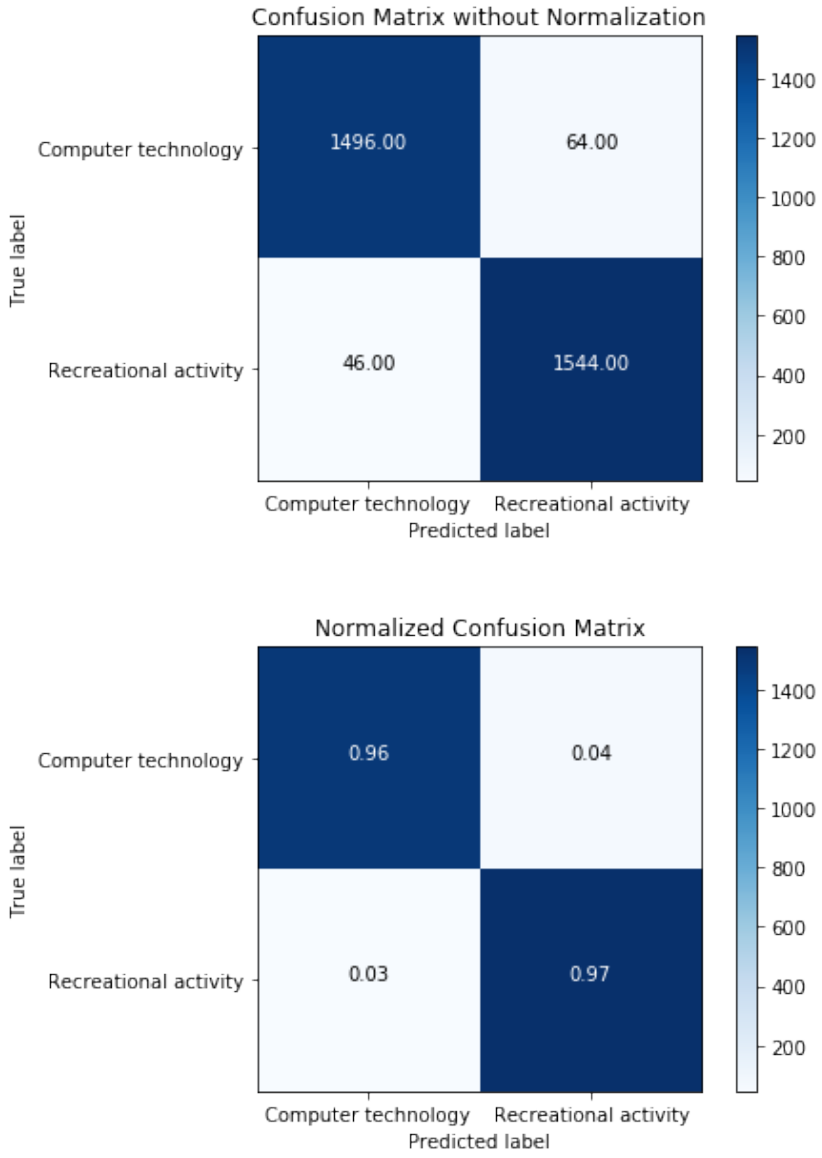
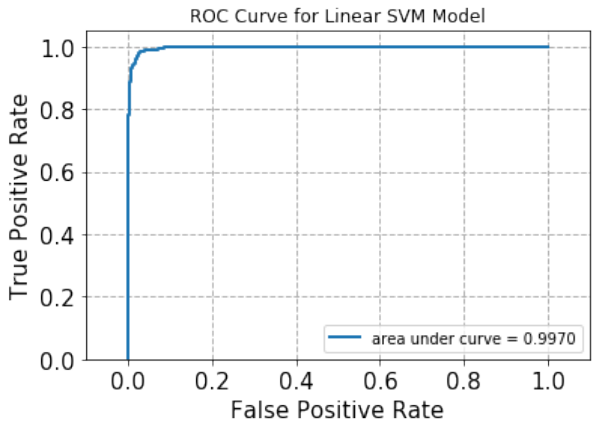


Figure 23: The confusion metric of l1 normal logistic regression with NMF & min_df = 2

————————————————RESULTS of l_2 normal logistic regression with LSI & min_df = 2



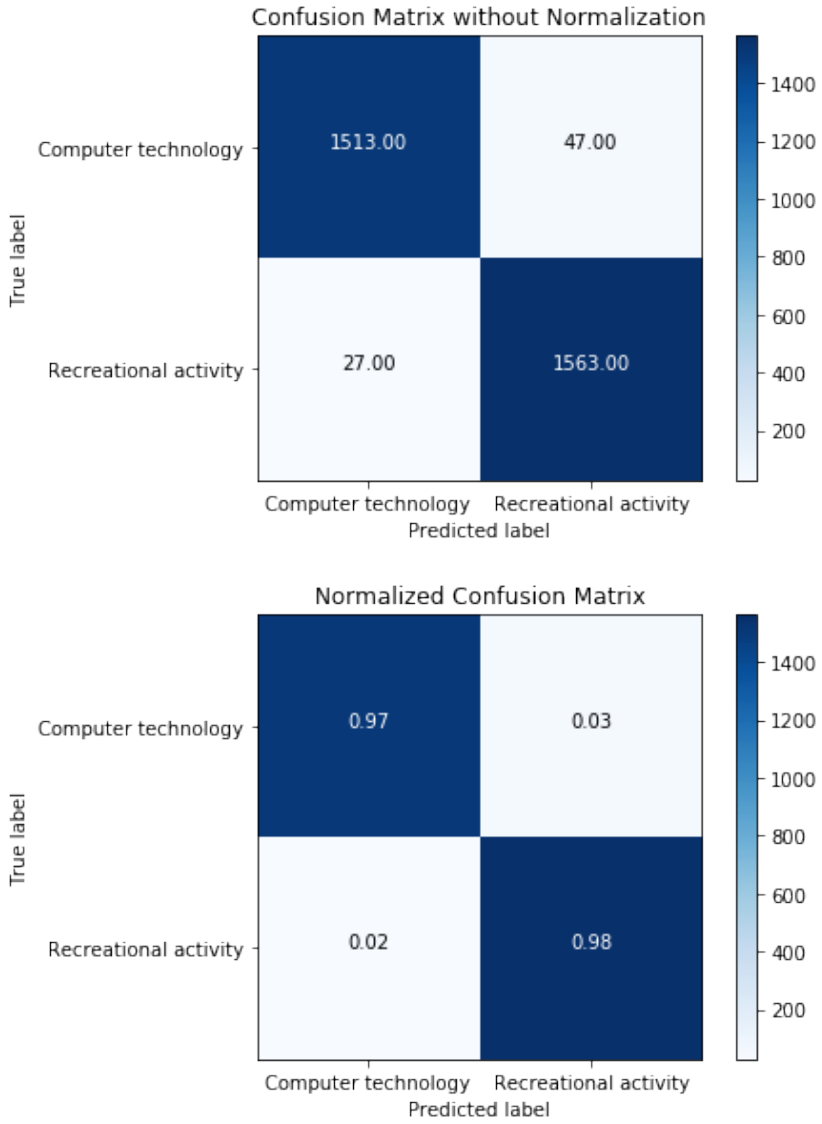
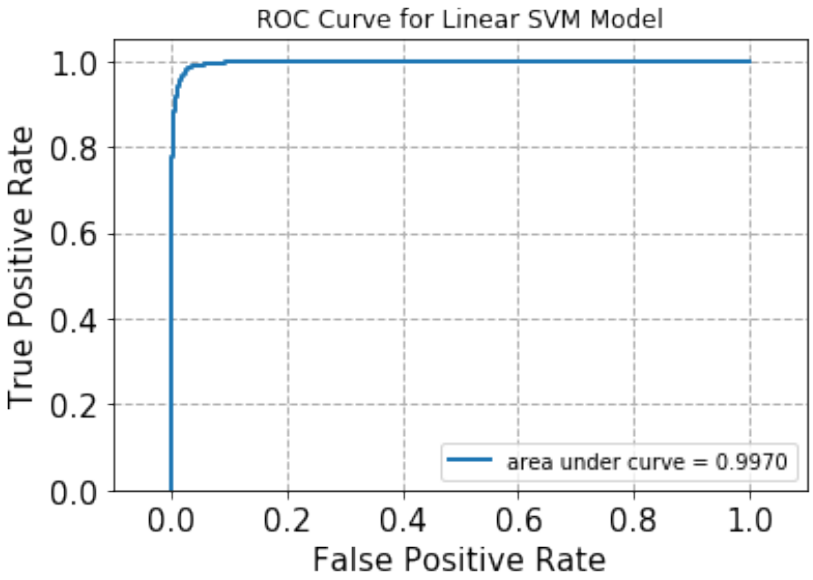
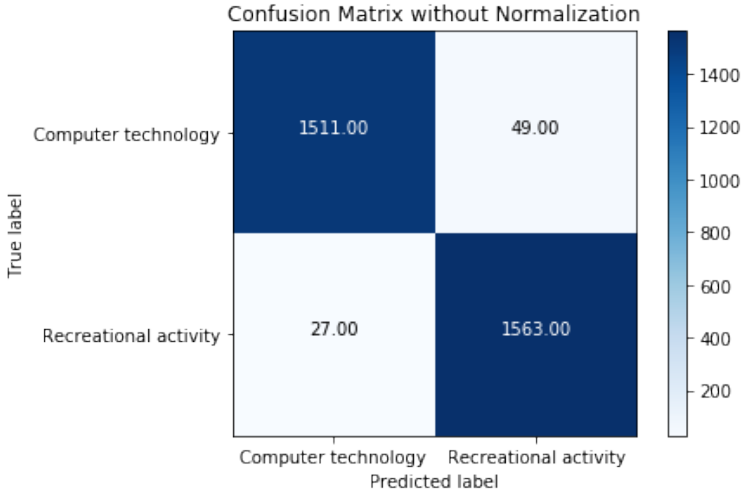


Figure 24: The confusion metric of l2 normal logistic regression with LSI & min_df = 2

————————————————RESULTS of l_2 normal logistic regression with LSI & min_df = 2





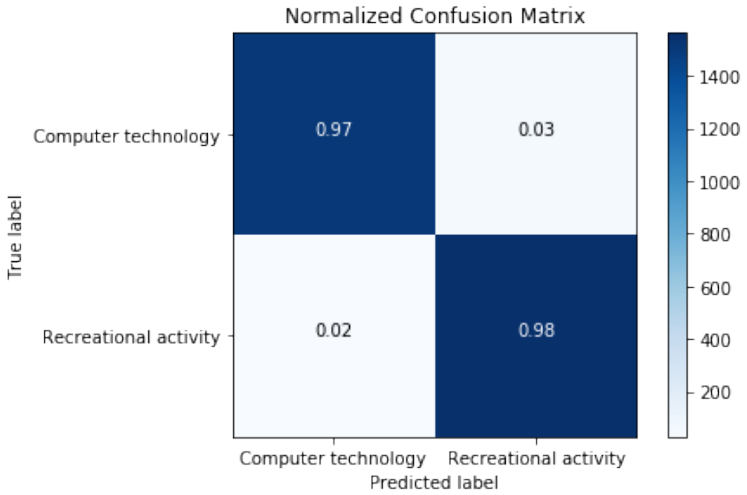


Figure 25: The confusion metric of l2 normal logistic regression with LSI & min_df = 5

————————————————RESULTS of l_2 normal logistic regression with NMF & min_df = 2

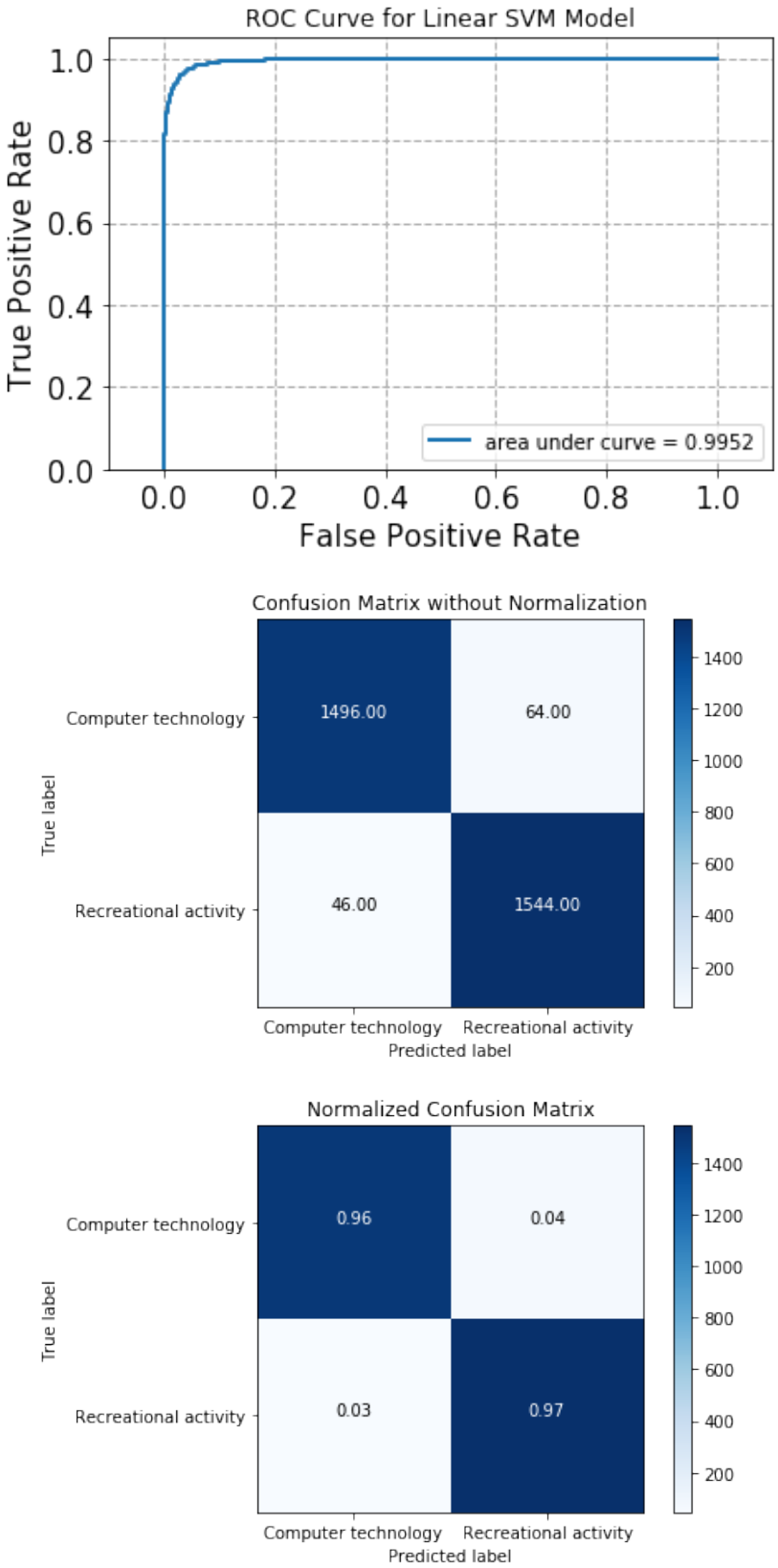


Figure 26: The confusion metric of l2 normal logistic regression with NMF & min_df = 2

Conclusion: Attached behind is the organized results of l_1 and l_2 logistic regression classifier in different situations, as shown in Table 6.

		LSI & min_df = 2	**LSI & min_df = 5**	**NMF & min_df = 2**
l_1 norm logistic regression	**Best regulation parameter**	100	10	100
	Evaluation coefficients with the best regulation parameter	Acc: 0.977 Rec: 0.983 Pre: 0.971 F1s: 0.977	Acc: 0.975 Rec: 0.983 Pre: 0.968 F1s: 0.976	Acc: 0.965 Rec: 0.971 Pre: 0.960 F1s: 0.966
l_2 norm logistic regression	**Best regulation parameter**	1000	100	10000
	Evaluation coefficients with the best regulation parameter	Acc: 0.977 Rec: 0.983 Pre: 0.971 F1s: 0.977	Acc: 0.976 Rec: 0.983 Pre: 0.970 F1s: 0.976	Acc: 0.965 Rec: 0.971 Pre: 0.960 F1s: 0.966

Table 6: The results of l1 and l2 logistic regression classifier in different situations

Reading the results from the Table 6, it can be observed that the evaluation coefficients obtained by l_1 norm logistic regression classifier and l_2 norm logistic regression classifier are roughly the same, all of which are above 0.96. It is sufficient to qualify both classifiers to be ideal for dealing with the data like this. Comparing the best regulation parameters obtained through the iteration, we found that different preprocessing methods as well as the value of min_df would affect the choice of best regulation parameters. Specifically, the best regulation parameters of l_2 norm logistic regression classifier are relatively larger than those of l_1 norm logistic regression classifier.

Multiclass Classification

Problem J

Requirement: In this part, we aim to learn classifiers on the documents belonging to the classes mentioned in part b; namely

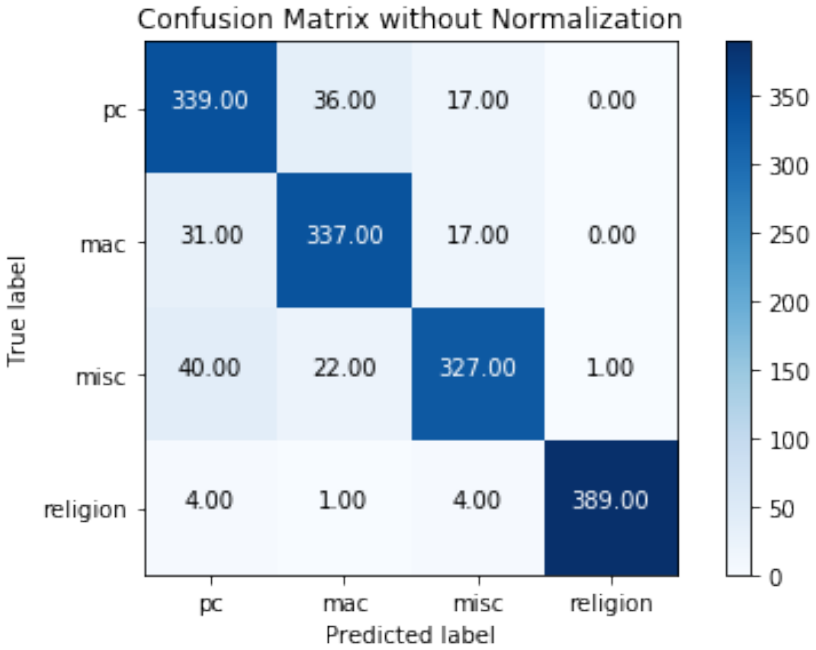
comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, misc.forsale, soc.religion.christian.

Perform Naïve Bayes classification and multiclass SVM classification (with both One VS One and One VS the rest methods described above) and report the confusion matrix and calculate the accuracy, recall and precision of your classifiers.

The classifiers we used above are all binary classification techniques. For Problem J, we are requested to use multiclass classification. And the steps do not change much from the binary classification. As for the naïve Bayes classification, we do the MultinomialNB as well as GaussianNB in order to compare these two classifiers and be echoed by the previous binary naïve Bayes classification. And we calculate the confuse matrix, accuracy, recall, precision and F1_score for every classification so as to compare the performances.

Result: The results of the Naïve Bayes classification (MultinomialNB and GaussianNB) and multiclass SVM classification (with both One VS One and One VS the rest methods described above) are listed behind, as shown in Figure 27-32.

————————————RESULTS of MultinomialNB classification



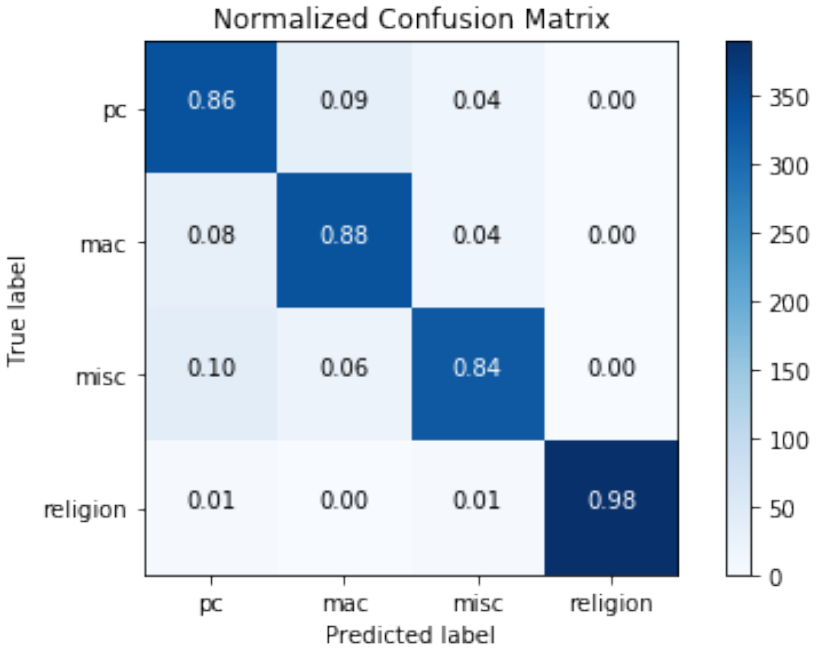
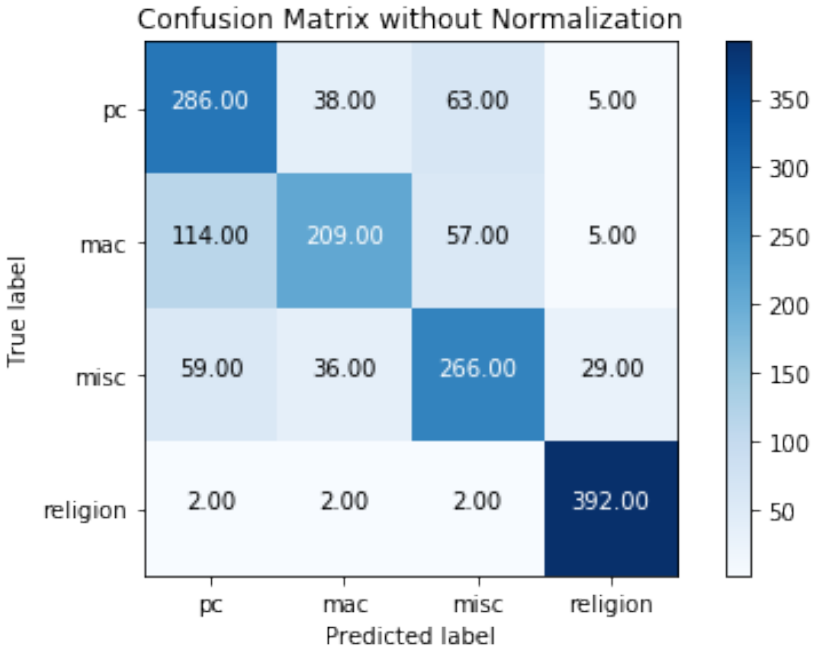


Figure 27: The confusion metric of MultinomialNB classification

————————————————RESULTS of GaussianNB classification



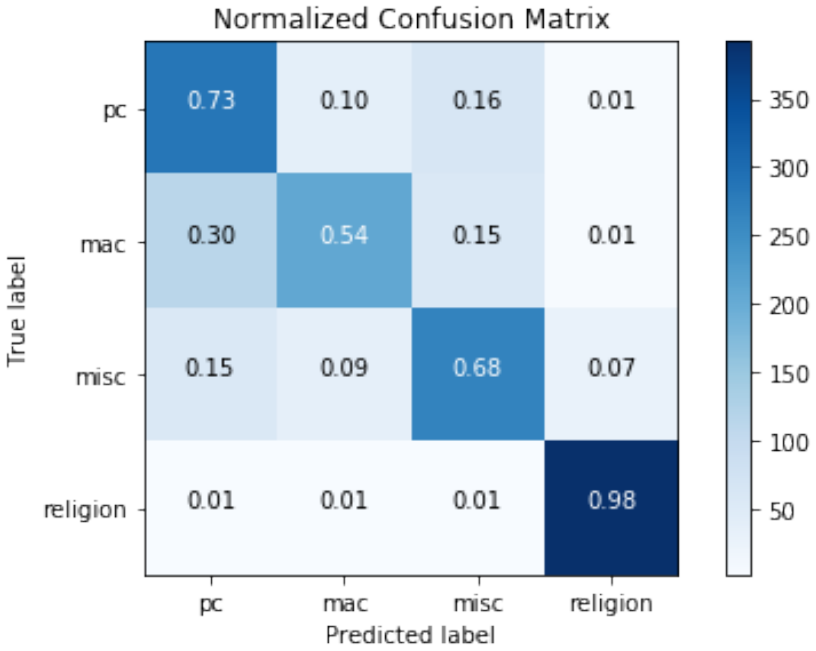
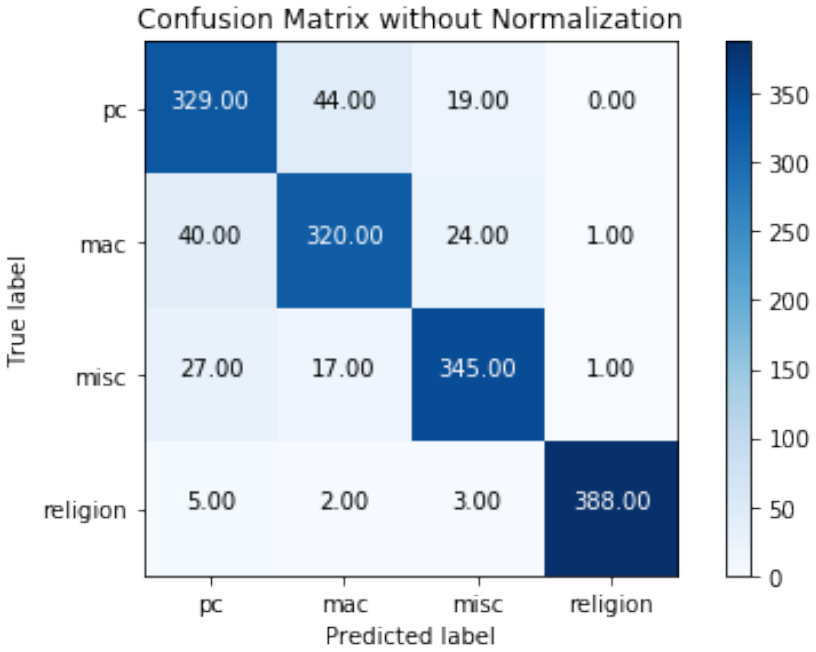


Figure 28: The confusion metric of Gaussian classification

————————————————RESULTS of Multiclass SVM (One VS One) with LSI



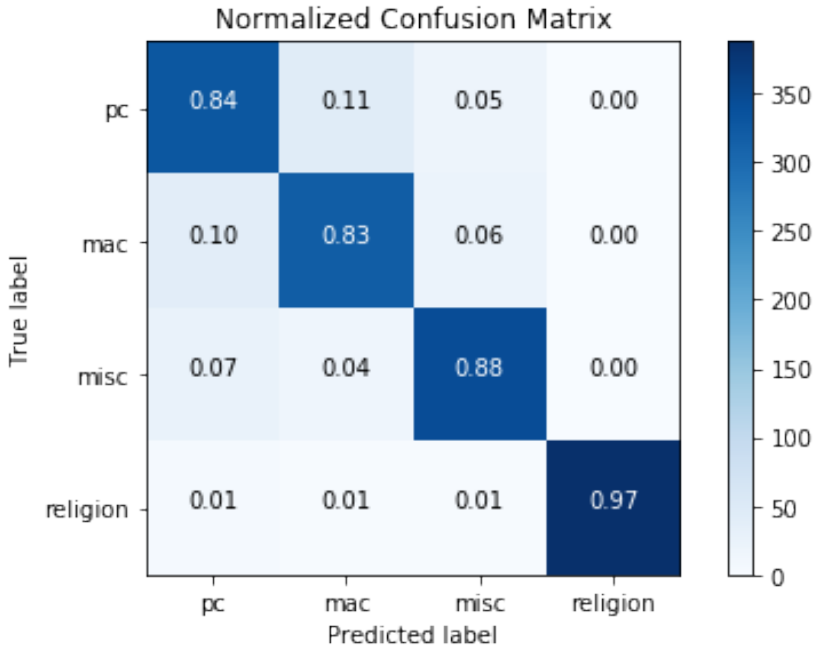
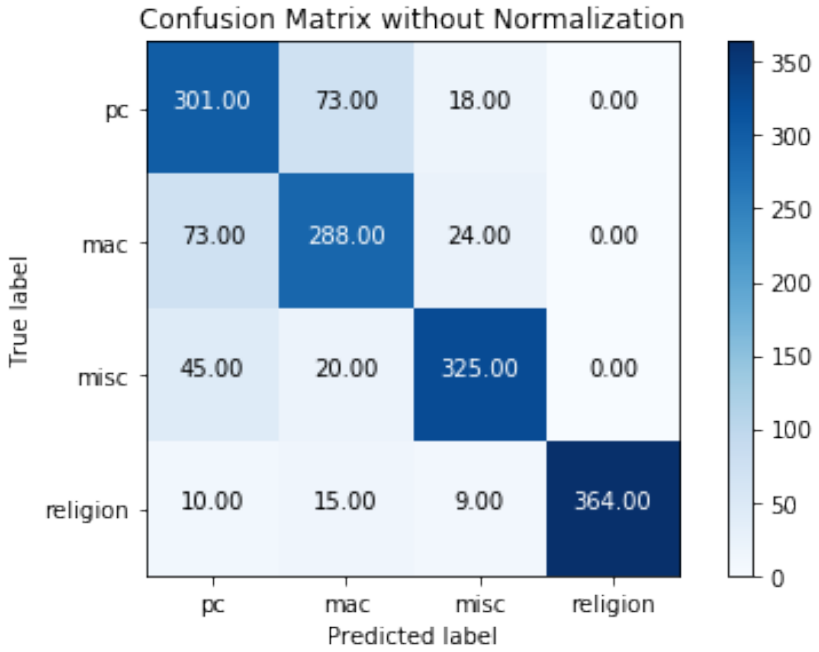


Figure 29: The confusion metric of Multiclass SVM (One VS One) with LSI

————————————————RESULTS of Multiclass SVM (One VS One) with NMF



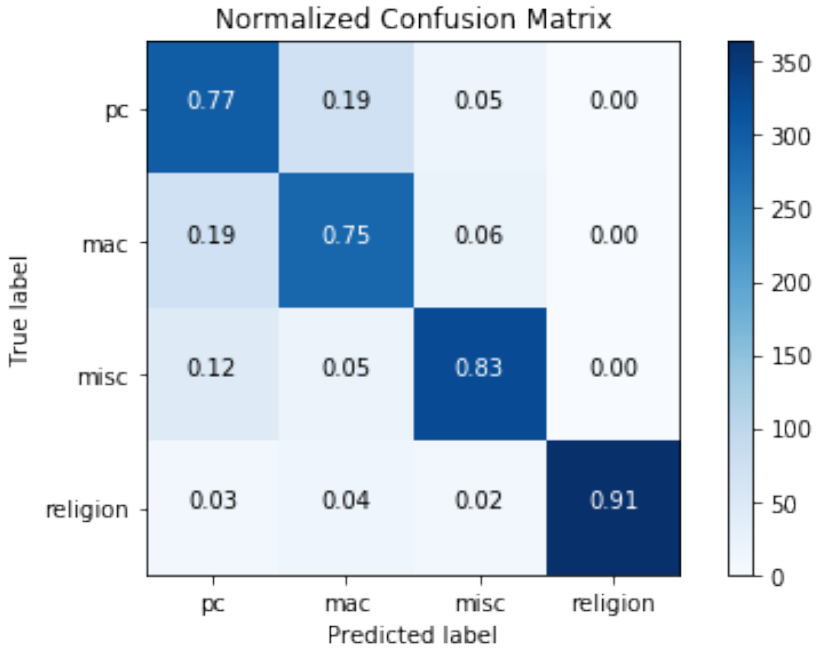
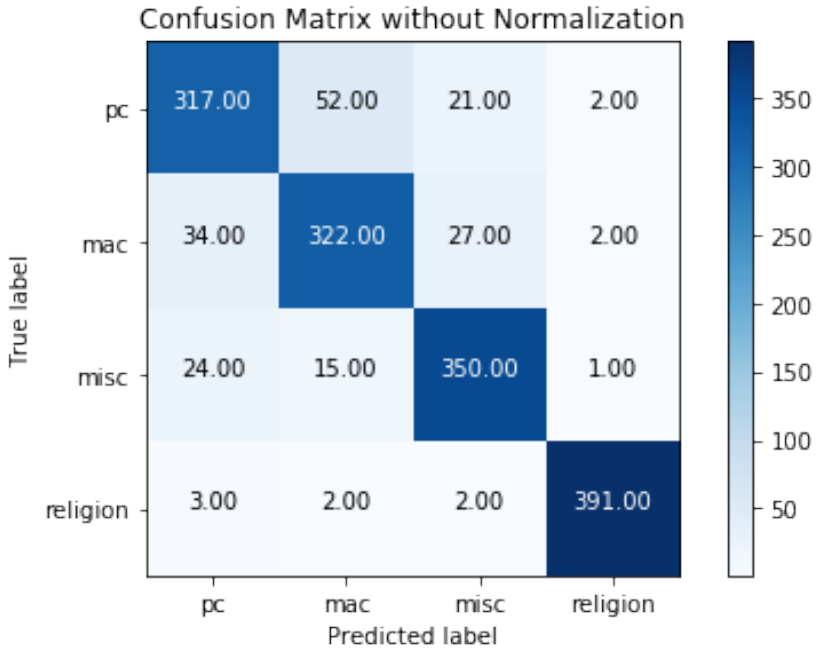


Figure 30: The confusion metric of Multiclass SVM (One VS One) with NMF

————————————————RESULTS of Multiclass SVM (One VS Rest) with LSI



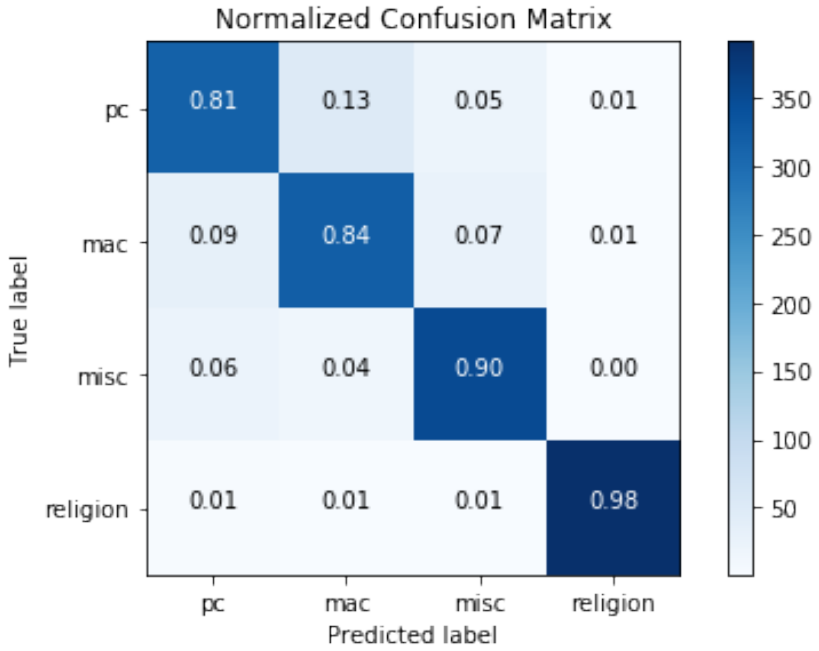
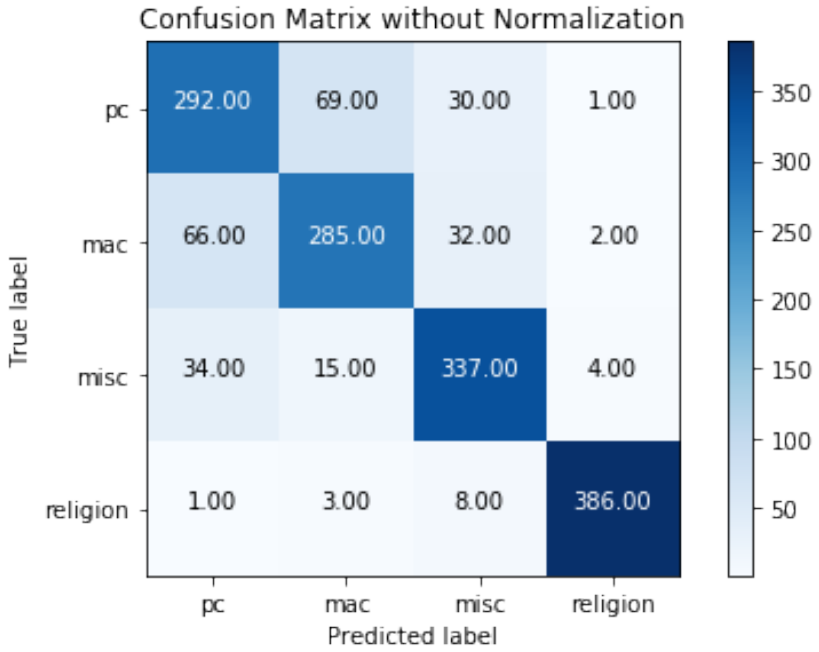


Figure 31: The confusion metric of Multiclass SVM (One VS Rest) with LSI

————————————————RESULTS of Multiclass SVM (One VS Rest) with NMF



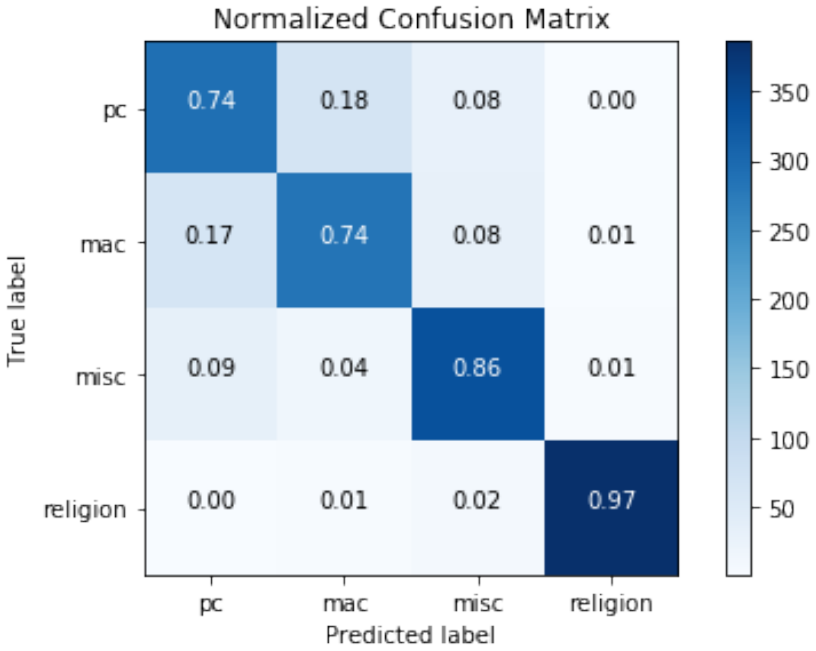


Figure 32: The confusion metric of Multiclass SVM (One VS Rest) with NMF

Conclusion: Attached behind is the organized results of, as shown in Table 7.

MultinomialNB	Acc: 0.889 Rec: 0.890 Pre: 0.891 F1s: 0.890	
GaussianNB	Acc: 0.737 Rec: 0.737 Pre: 0.738 F1s: 0.737	
	LSI & min_df=2	**NMF & min_df=2**
One VS One Multiclass SVM	Acc: 0.883 Rec: 0.883 Pre: 0.884 F1s: 0.884	Acc: 0.817 Rec: 0.817 Pre: 0.824 F1s: 0.820

One VS Rest Multiclass SVM	Acc: 0.882 Rec: 0.882 Pre: 0.882 F1s: 0.882	Acc: 0.831 Rec: 0.831 Pre: 0.831 F1s: 0.831

Table 7: The results of naïve Bayes and Multiclass classifier in different situations

From the organized table results, it is most obvious that MultinomialNB classification model has better performance than the GaussianNB classification model, which is similar to the results in the binary classification. However, if we compare the results of the binary classification and the multiclass classification, the multiclass classification works not so perfectly as the binary classification. And it is not difficult to understand that multiclass increases the difficulty for the classifier to predict.

When we focus on the One VS One Multiclass SVM, the classifier when using LSI to reduce dimension is better than that using NMF as dimension reduction. The situation is similar when we focus on the One VS Rest Multiclass SVM. And both using LSI, the One VS One Multiclass SVM works better than One VS Rest Multiclass SVM, while both using NMF the One VS One Multiclass SVM works worse than One VS Rest Multiclass SVM.