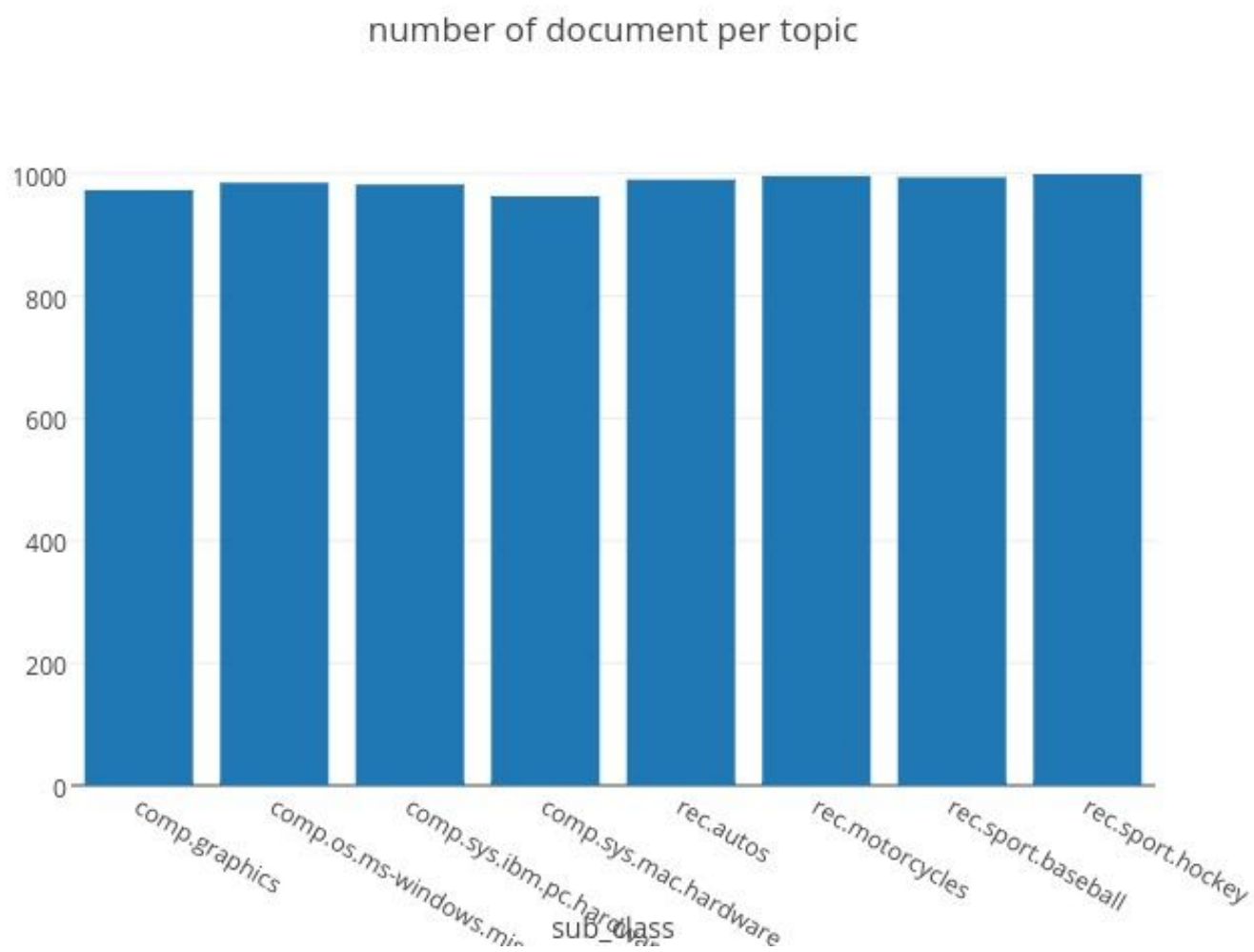
EE219 Project 2
Classification Analysis

Winter 2017

Boyang Cai 304330123
Manni Chen 304145309
Zhuoqi Li 004855607

Part A. Histogram Plotting

The histogram plot of the number of documents per topic is shown as below:



The number of documents in **Computer Technology** is **3903**.
The number of documents in **Recreational Activity** is **3979**.

Part B. Data Preprocessing and TFxIDF representation

Before we start to do the classification, the primary step in classifying a corpus of text is document representation. In order to capture the importance of a word (term) to a document, we use the Term Frequency-Inverse Document Frequency (TFxIDF) metric. This metric is measured to reveal not only the normalized words count in the document but also the frequency of terms in the corpus. The TFxIDF matrix obtained from this part of the project will be later used as the input to perform classification on.

Before we process the data to get the TFxIDF matrix, we need to perform a preprocess on the terms in every document. Because of the nature of English language, we know that stop words will appear frequently in any document regardless of their topics and therefore will mask the term frequency we really care about. Therefore, we need to take them out before generating the matrix. Same thing apply to stem words, which will introduce confusion in classification analysis.

After taking out stem and stop words, we get the following results on **all 20 categories**:
Term number: 79888
Document number: 11314

For **Computer technology** and **Recreational activity** only:
Term number: 50492
Document number: 4732

We will use only perform classification analysis on the above two fields only in later parts.

Part C. TFxICF 10 most significant terms

In this part, instead of using the TFxIDF, we use Term Frequency Inverse Category Frequency (TFxICF) metric. Similar to the TFxIDF approach, TFxICF shows the terms frequency against each class. The TFxIDF matrix has documents as its row and terms as its column. For TFxICF, now each row represents a category and there are 20 rows. By applying this metric, we could have an idea what are the most frequently used terms in each category.

Here are the 10 most frequently used terms in ***comp.sys.ibm.pc.hardware***, ***comp.sys.mac.hardware***, ***misc.forsale*** and ***soc.religion.christian*** category (Increasing order, least -> most):

Comp.sys.ibm.pc.hardware: *bio**, system, bus, ide, use, control, disk, card, scsi, drive*
Comp.sys.mac.hardware : *bit**, monitor, card, simm, problem, scsi, use, drive, appl, mac*
Misc.forsale: *game, condit, sell, price, includ, new, ship, dos, offer, sale*
Soc.religion.christian: *say, would, believ, faith, one, christ, church, jesus, christian, god*

The terms in lists above does comply with what we expect to be in each category, which demonstrate the effectiveness of the TFxICF metric.

Part D. Latent Semantic Indexing dimension reduction

Although TFxIDF could be a good metric, but its dimension (4732, 50492) is too large to perform any classification algorithm efficiently. Therefore, we use Latent Semantic Indexing (LSI), a dimension reducing transform that finds the optimal representation of the data in a lower dimension space in the mean squared error sense.

According to the spec, we choose k to be 50 - reduce the TFxIDF matrix from (4732, 50492) to (4732, 50). In order to do this, we use the 'TruncatedSVD' method from 'sklearn' package. We knew that before the number 50492 stands for the total distinct terms appear in chosen 8 categories of documents. So what is the physical meaning of the number '50' here? We do not know, but we do know that they are the lower dimension projections of these 50492 terms.

Part E. Learning algorithm: Linear Support Vector Machines

Now with the reduced-dimension TFxIDF matrix as the train input we could start to apply the classification learning algorithm and so prediction. The first algorithm we are going to use is the Linear Support Vector MAchines which classifies data with a linear hyperplane with max margin. This algorithm fits our purpose of classifying files into two classes: computer hardware and recreational events. We divided the dataset into the train and test set. We use the reduced-dimension matrix TFxIDF of the train set and their known categories to fit the model and apply this model on the reduced-dimension test TFxIDF for prediction.

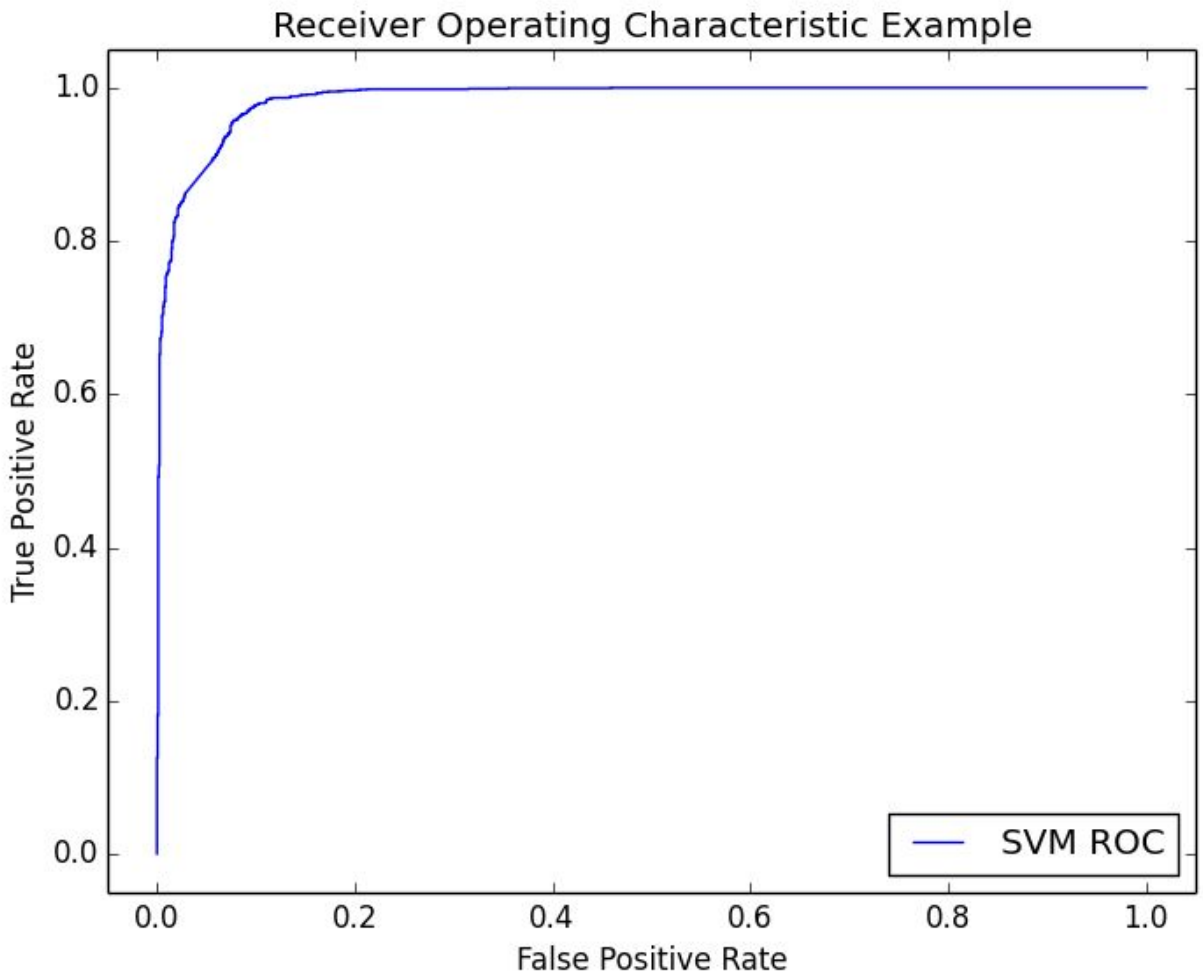
This following **accuracy summary** is generated using the 'metric.classification_report' method of sklearn:

```
The accuracy for the model is 0.939365
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:
             precision    recall  f1-score   support

          0       0.95      0.92      0.94      1560
          1       0.93      0.96      0.94      1590

avg / total       0.94      0.94      0.94      3150
```

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1439	121
Recreation actual	70	1520

The following **ROC curve** is generated using ‘metrics.roc_curve’ method:



Part F. Learning algorithm: Soft Margin SVM with different C(gamma)

After we tried with the Part e, we notice that the method we use is a hard SVM which does not have any tolerance on the mismatch data. However, there could be some outliers in the real time so we should consider the error into our learning method as well.

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

In the above formula, we add the part where the slack variables could be eliminated from the function with a coefficient of gamma.

We got the following data for each of the gamma we choose from [0.001, 0.01, 0.1, 1, 10, 100, 1000]:

****

The accuracy for the model under gamma 0.001 is 0.504761904762
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	N/A	0.00	0.00	1560
1	0.50	1.00	0.67	1590
avg / total	0.25	0.50	0.34	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	0	1560
Recreation actual	0	1590

****

The accuracy for the model under gamma 0.01 is 0.914603174603
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support

0	0.89	0.94	0.92	1560
1	0.94	0.89	0.91	1590
avg / total	0.92	0.91	0.91	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1472	88
Recreation actual	181	1409

****

The accuracy for the model under gamma 0.1 is 0.935555555556
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.94	0.93	0.93	1560
1	0.93	0.94	0.94	1590
avg / total	0.94	0.94	0.94	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1447	113
Recreation actual	90	1500

****

The accuracy for the model under gamma 1 is 0.939682539683
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.96	0.91	0.94	1560
1	0.92	0.97	0.94	1590

avg / total 0.94 0.94 0.94 3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1423	137
Recreation actual	53	1537

****

The accuracy for the model under gamma 10 is 0.938095238095
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.96	0.92	0.94	1560
1	0.92	0.96	0.94	1590
avg / total	0.94	0.94	0.94	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1431	129
Recreation actual	66	1524

****

The accuracy for the model under gamma 100 is 0.938095238095
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.96	0.92	0.94	1560
1	0.92	0.96	0.94	1590
avg / total	0.94	0.94	0.94	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1432	128
Recreation actual	67	1523

****

The accuracy for the model under gamma 1000 is 0.93746031746
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.95	0.92	0.94	1560
1	0.92	0.96	0.94	1590
avg / total	0.94	0.94	0.94	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

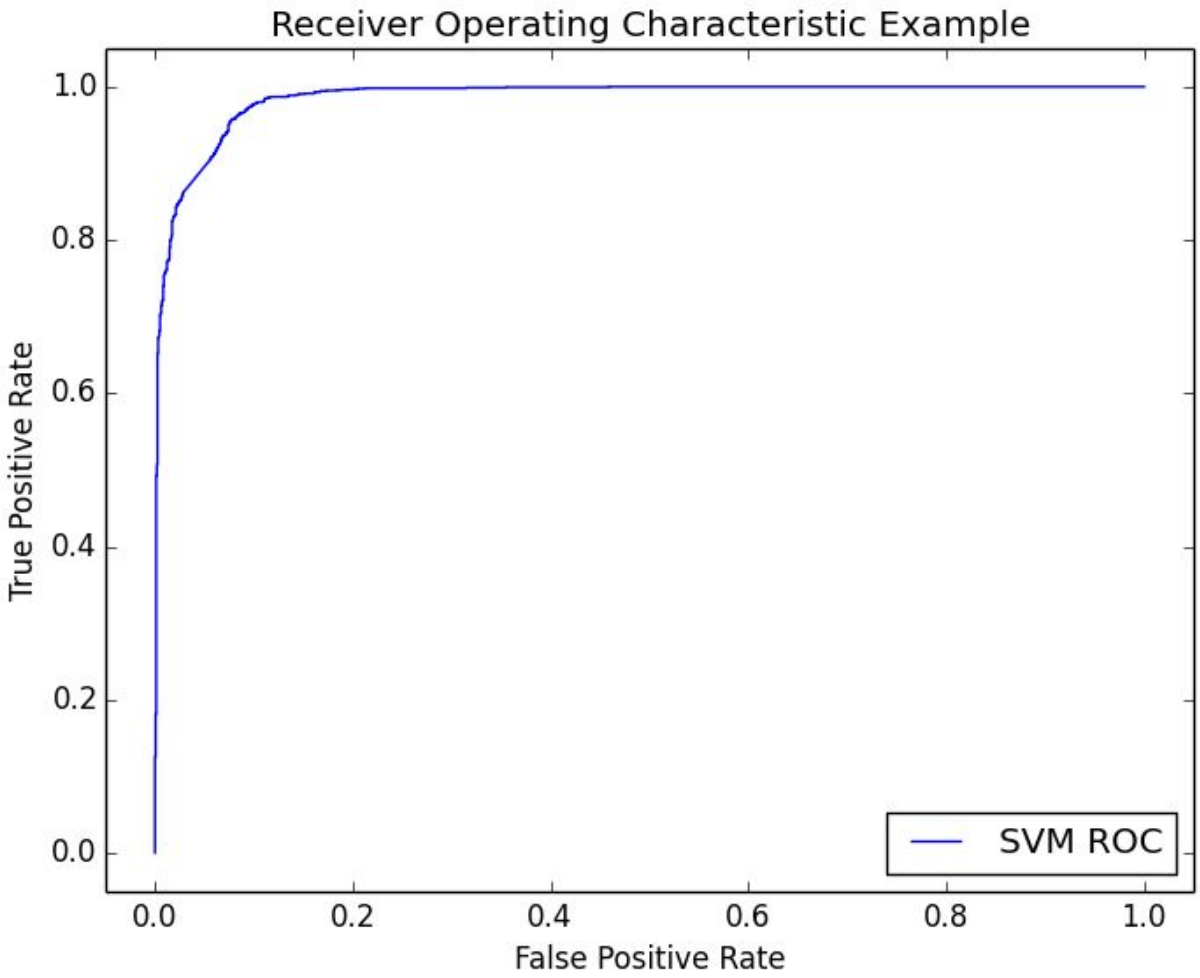
	Computer predict	Recreation predict
Computer actual	1433	127
Recreation actual	70	1520

The accuracy according to each gamma is:

Gamma	0.001	0.01	0.1	1	10	100	1000
Accuracy	0.504761904762	0.914603174603	0.935555555556	**0.939682539683**	0.938095238095	0.938095238095	0.937460317460

According to the data, we got that the highest Accuracy is 0.939682539683 at gamma = 1

The following **ROC curve** is generated at the value of gamma = 1 using ‘metrics.roc_curve’ method:



As we compared the data with part E, the accuracy of the learning algorithm with no error tolerance is 0.939365. It is slightly lower than the soft SVM with tolerance gamma = 1, which is 0.939682539683.
the confusion matrix for part E is :

	Computer predict	Recreation predict
Computer actual	1439	121
Recreation actual	70	1520

where the optimal confusion matrix for F is:

	Computer predict	Recreation predict
Computer actual	1423	137
Recreation actual	53	1537

We can see that with an increasing number of gamma, there is only a slightly difference in the accuracy and the confusion matrix. However, soft SVM does have a increase on the accuracy of the result.

Part G. Multinomial Naive Bayes Algorithm Classification

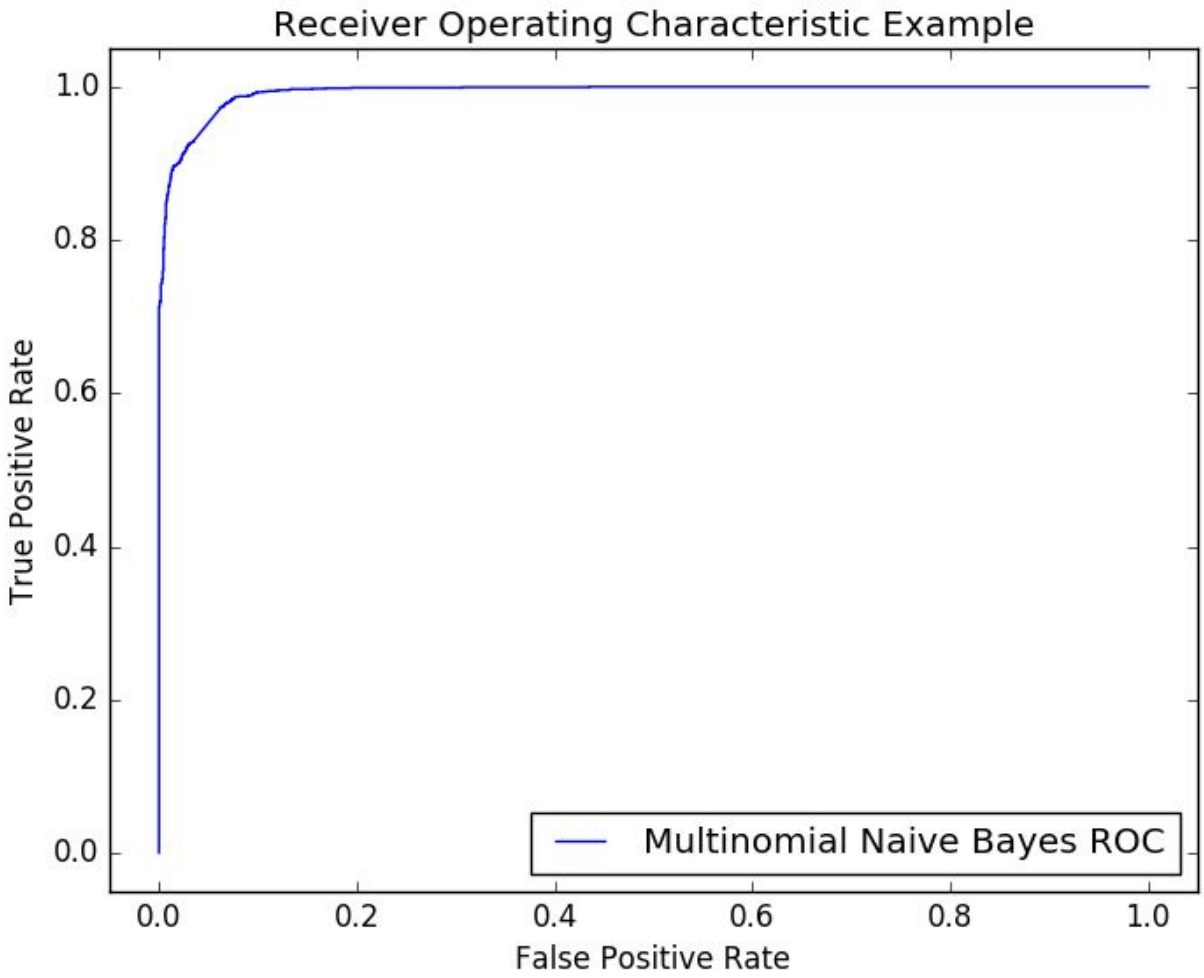
The accuracy for the multinomial naive bayes model is **0.955238**
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.97	0.94	0.95	1560
1	0.94	0.97	0.96	1590
avg / total	0.96	0.96	0.96	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1462	98
Recreation actual	43	1547

The following **ROC curve** is generated using ‘metrics.roc_curve’ method:



Part H. Logistic Regression Classifier

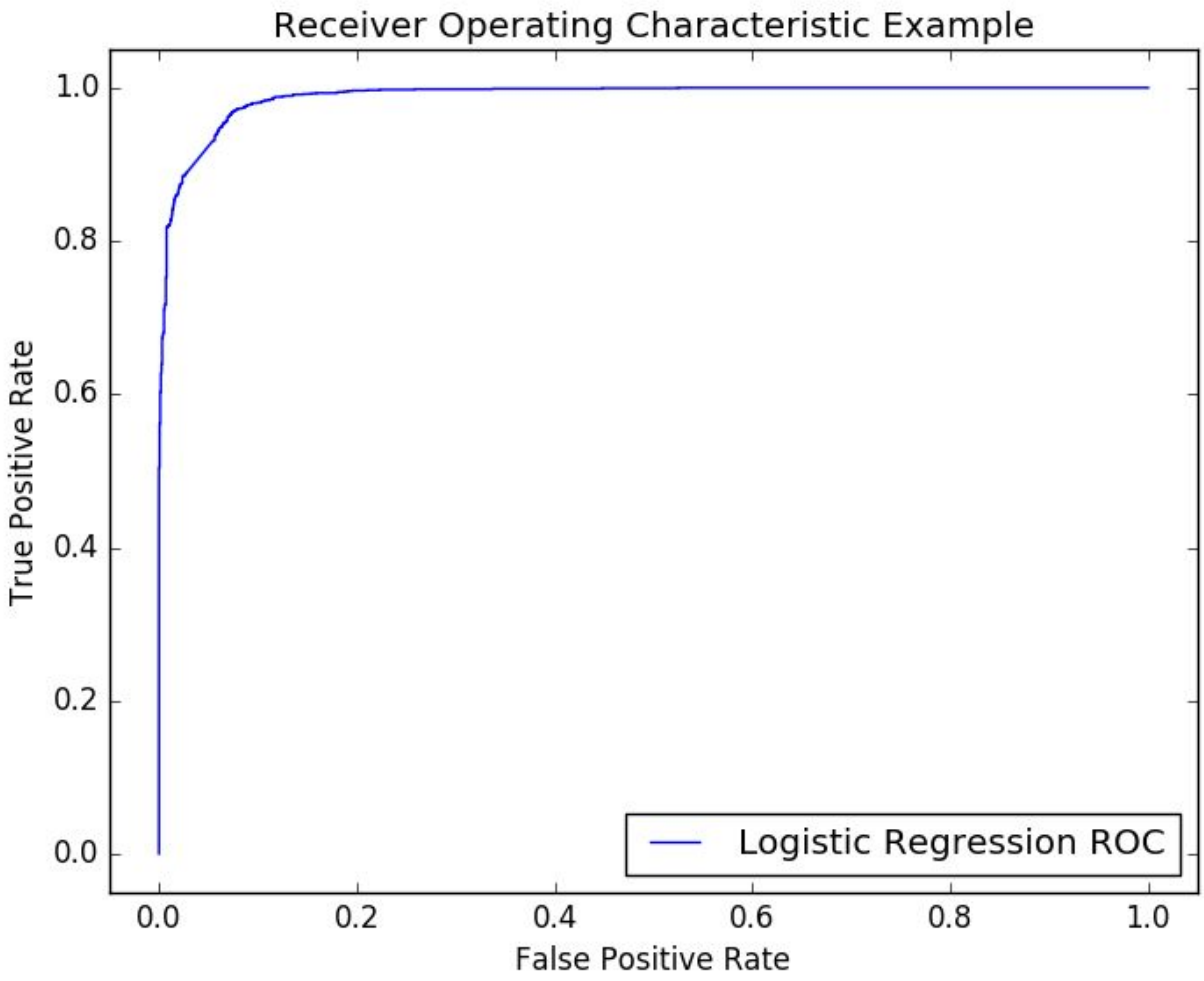
The accuracy for the logistic regression classifier model is **0.942857**
'0' is Computer Technology and '1' is Recreational Activity
The precision and recall values are:

	precision	recall	f1-score	support
0	0.98	0.91	0.94	1560
1	0.91	0.98	0.95	1590
avg / total	0.95	0.94	0.94	3150

The following **confusion matrix** is generated using ‘metric.confusing_matrix’ method of sklearn:

	Computer predict	Recreation predict
Computer actual	1414	146
Recreation actual	34	1556

The following **ROC curve** is generated using ‘metrics.roc_curve’ method:



Part I. Logistic Regression with L1/L2 Regularization

Here is the accuracy comparison for logistic regression with L1/L2 regularization and different coefficients

	C = 100	C = 1	C = 0.01
L1 Regularization	0.938413	0.906349	0.495238
L2 Regularization	0.943175	0.94254	0.89873

The Coefficients of the fitted hyperplane via L1/L2 regularization in different regulation coefficients

	C = 100	C = 1	C = 0.01
L1 Regularization	0.983727810651	0.927726120034	0.495139475909
L2 Regularization	0.983727810651	0.969146238377	0.913989856298

According to the tables above, when the regularization coefficients goes small (100 -> 0.01), accuracy for both regularization models decreases. Also, the coefficients of the fitted hyperplane decrease. The change of regularization coefficient has a greater influence on L1 regularization than that on L2. When the number of irrelevant features grows logarithmically, it is better to use L1 regularization with logistic regression and when the number of irrelevant features grows linearly, it is better to use L2 regularization.

Part J. SVM with OVO, OVR and Naive Bayes Comparison

The accuracy for the **SVM_OVO** model is **0.818530351438**
'0' is comp.sys.ibm.pc.hardware, '1' is comp.sys.mac.hardware, '2' is misc.forsale and '3' is soc.religion.christian
The precision and recall values are:

	precision	recall	f1-score	support
0	0.73	0.77	0.75	392
1	0.73	0.75	0.74	385
2	0.86	0.82	0.84	390
3	0.96	0.93	0.95	398
avg / total	0.82	0.82	0.82	1565

The confusion matrix is as shown below:

```
[[300  63  26   3]
 [ 69 290  21   5]
 [ 35  27 321   7]
 [  7  15   6 370]]
```

The accuracy for the **SVM_OVR** model is **0.818530351438**
'0' is comp.sys.ibm.pc.hardware, '1' is comp.sys.mac.hardware, '2' is misc.forsale and '3' is soc.religion.christian
The precision and recall values are:

	precision	recall	f1-score	support
0	0.73	0.77	0.75	392
1	0.73	0.75	0.74	385
2	0.86	0.82	0.84	390
3	0.96	0.93	0.95	398
avg / total	0.82	0.82	0.82	1565

The confusion matrix is as shown below:

```
[[300  63  26   3]
 [ 69 290  21   5]
 [ 35  27 321   7]
 [  7  15   6 370]]
```

The accuracy for the **NB** model is **0.747603833866**
'0' is comp.sys.ibm.pc.hardware, '1' is comp.sys.mac.hardware, '2' is misc.forsale and '3' is soc.religion.christian
The precision and recall values are:

	precision	recall	f1-score	support
0	0.61	0.74	0.67	392
1	0.70	0.61	0.65	385
2	0.82	0.71	0.76	390
3	0.88	0.93	0.91	398
avg / total	0.75	0.75	0.75	1565

The confusion matrix is as shown below:

```
[[291  67  23  11]
 [105 233  25  22]
 [ 70  28 276  16]
 [ 10   6  12 370]]
```