Project 2 - Classification Analysis

Brendon Faleiro - 704759004
Anurag Pande - 604749647
Sachin Bhat - 304759727
February 16, 2017

1 Introduction

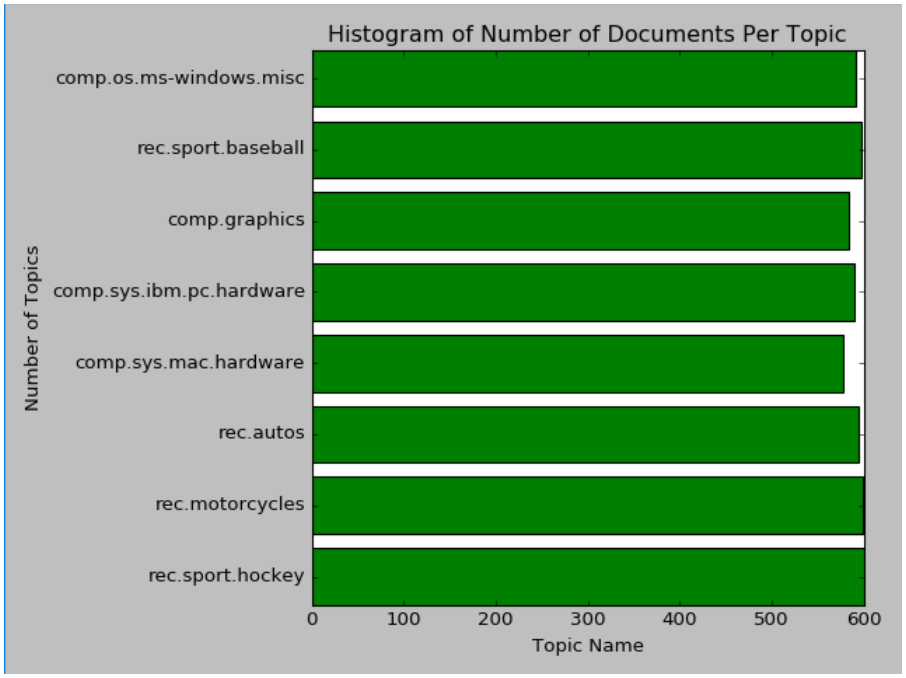
The main aim of this project is to understand the performance of different classification techniques on the same dataset to have a definitive comparison. The dataset used for all the comparisons is the 20 Newsgroups dataset, which is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups.

The classifiers analyzed are Naive Bayes, Support Vector Machines and Logistic Regression. We also look at One vs Many and One vs Rest methods of multiclass classification using Naive Bayes and SVM. The objective of this project is to gain better insight into these classification methodologies and, specifically, how to correctly preprocess and classify textual data into predefined classes, given training data.

2 Dataset and Problem Statement

2.1 Question (a) - Histogram of documents versus topics

A dataset is said to be balanced if the distribution of data in each class is approximately equal. Unbalanced datasets must be handled properly, and is difficult to classify without certain necessary transformations. So, to make sure the distribution is more or less even, we plotted the histogram of the number of documents per topic.



Number of documents in:

- Computer Technology
  - Training Dataset : 2343
  - Testing Dataset : 1560
- Recreational Activity
  - Training Dataset : 2389
  - Testing Dataset : 1590

3 Modeling Text Data and Feature Extraction

3.1 Question (b) - Pre-processing and TF-IDF Representation

Representation of the document should be succinct to avoid overfitting, but not leave out essential features. Thus extremely common words (stop words and non ASCII characters) are removed from the vocabulary, along with very rare ones. This avoids unnecessarily large feature vectors. Python's NLTK was used to for this purpose.

After preprocessing, we find the TFxIDF representation of the document using a transformer, giving us the number of terms in each document.

Number of Terms Extracted = 25462

3.2 Question (c) - 10 most Significant Terms

To find the top 10 most significant terms per class, the following steps were taken:

1. Documents were first preprocessed (stop words and punctuation removal, stems).
2. Unique terms were mapped to their count.
3. Using the given formula, TFxICF was calculated for each term.

comp.sys.ibm.pc.hardware	**comp.sys.mac.hardware**	**misc.forsale**	**soc.religion.christian**
adaptec	iisi	obo	liturgi
motherboard	duo	hobgoblin	kulikauska
irq	quadra	spiderman	clh
vlb	centri	liefeld	christ
aspi	powerbook	hiram	atheist
dx	nubus	xforc	cathol
floppi	fpu	hulk	atho
scsi	scsi	sabretooth	sabbath
ide	lciii	wolverin	resurrect
jumper	simm	forsal	scriptur

4 Feature Selection

4.1 Question (d) - LSI Decomposition of TFxIDF Matrix

To increase the performance of any classifier, dimensionality of the data must be reduced, usually by selecting a non-sparse subset of the total features. We've used Latent Semantic Indexing (LSI), which reduces dimensionality using mean-squared errors.

The principle behind LSI is that some words that are used in the same contexts usually have meanings that are similar. We reduce the features to lower dimensional space by representing data in term document matrix, with columns of TFxIDF representation of documents that we got in question (c). Steps followed:

1. Preprocess training and testing datasets
2. Calculate TFxIDF vector representation
3. Prune features using LSI Decomposition with k = 50

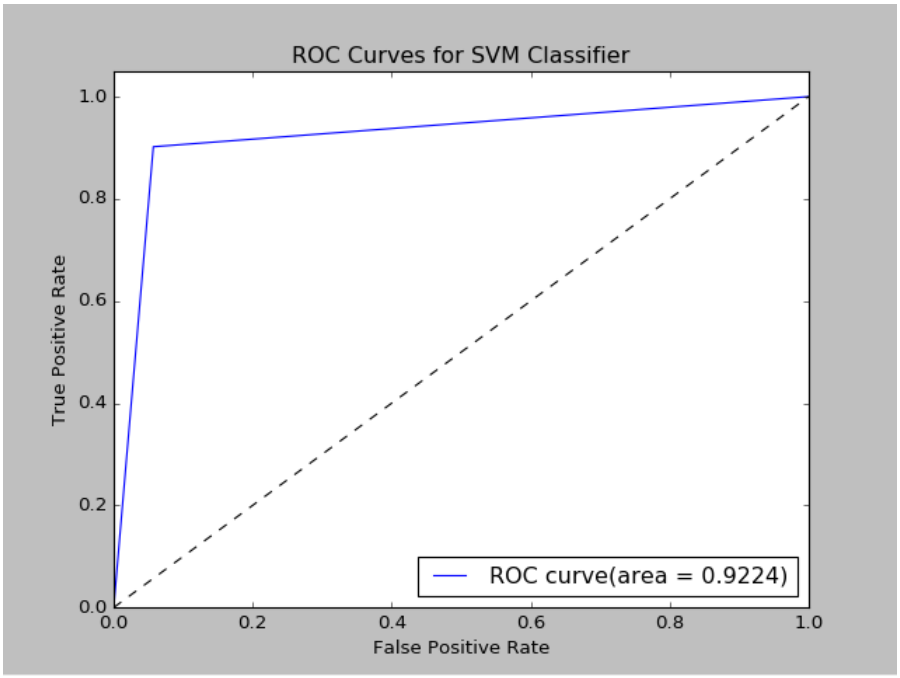
5 Learning Algorithms

5.1 Question (e) - Linear Support Vector Machines

Linear SVM classifiers work by interpreting the sign of the vector representation of the document multiplied by a weight. A positive sign means a document belongs to one class, while a negative sign means it belongs to the other class. We used a linear kernel to train out classifier. We then ran it on our training dataset. The statistics obtained for the classifier are:

Statistic	Result
Accuracy	92.222
Precision	94.156
Recall	90.189

	Predicted: Computer	Predicted: Recreational
Actual: Computer	1471	89
Actual: Recreational	156	1434



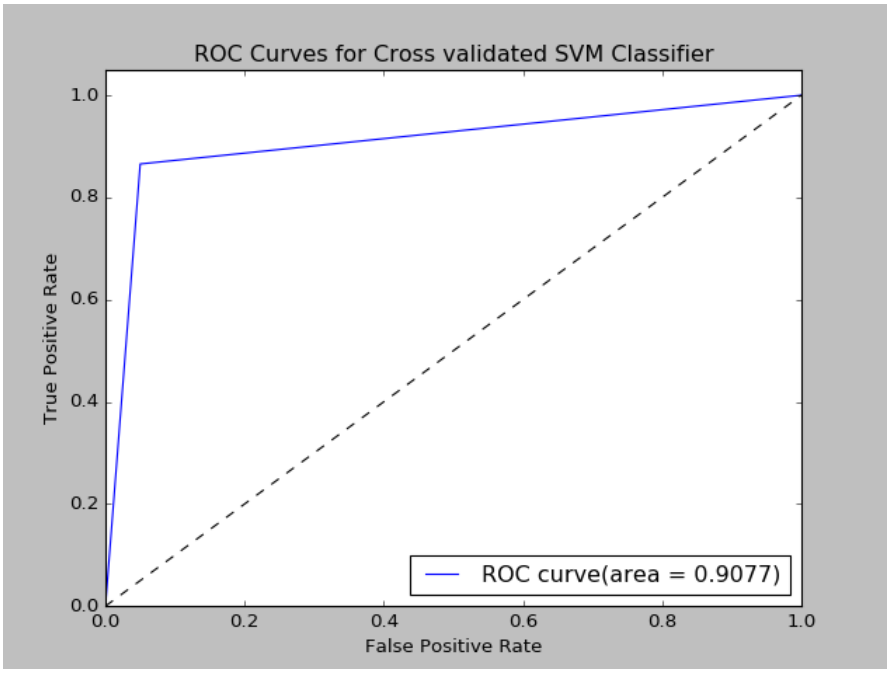
The Receiver Operating Characteristic (ROC) curve is created by plotting the true positive rate (TPR) versus the false positive rate (FPR) at multiple thresholds. An area of 1 signifies perfect classification. The ROC we obtains shows all the classes have an area approximately equal to 1. Thus, we can safely say that all our test cases are correctly classified.

5.2 Question (f) - Soft Margin Support Vector Machines

We then used soft-margin SVM to minimize training error, followed by a 5-fold cross validation. We found that the value where Soft-SVM gave the best results was at k = 0??? which is equivalent to regular SVM. The classifier statistics are listed below:

Statistic	Result
Accuracy	90.730
Precision	94.635
Recall	86.540

	Predicted: Computer	Predicted: Recreational
Actual: Computer	1482	78
Actual: Recreational	214	1376

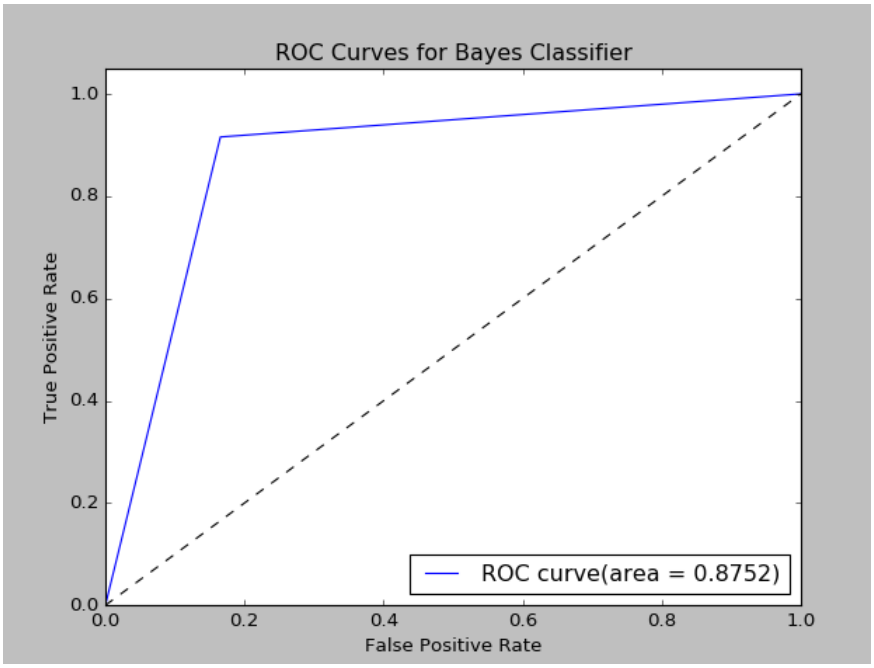


5.3 Question (g) - Naive Bayes

Ques (g) Naive Bayes Now, we turn to Naive Bayes to perform the same tests as in question (f). 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. We used a Gaussian Naive-Bayes classifier, the test statistics for which are:

Statistic	Result
Accuracy	87.550
Precision	84.950
Recall	91.570

	Predicted: Computer	Predicted: Recreational
Actual: Computer	1302	258
Actual: Recreational	134	1456



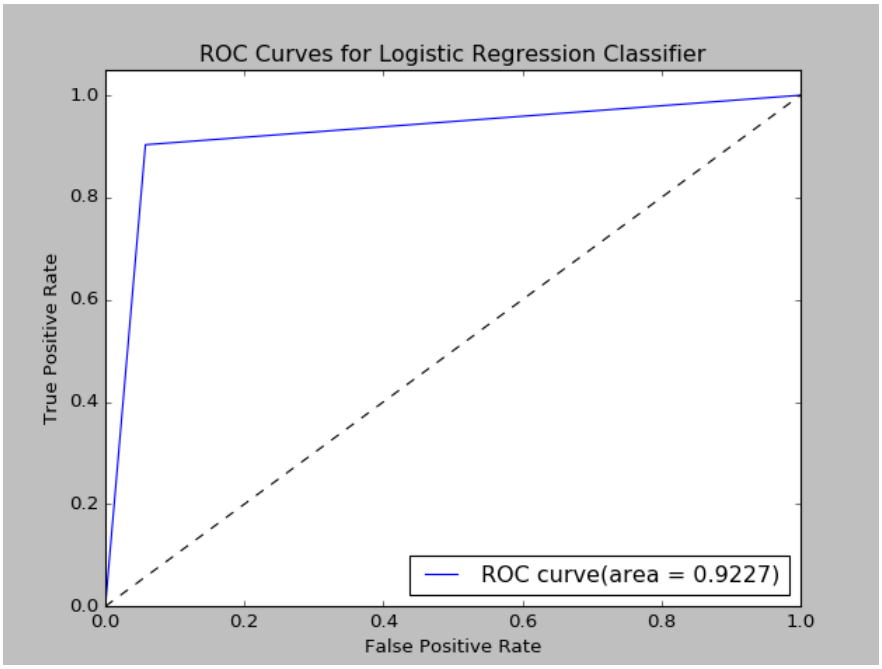
As seen above Naive Bayes has noticeably less area under the ROC curve as compared to SVM classifier signifying that there are more records that were incorrectly classified as compared to SVM classifier.

5.4 Question (h) - Logistic Regression

The same tests are now applied to Logistic Regression (LR). LR quantifies the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. The statistics obtained for the LR classifier are:

Statistic	Result
Accuracy	92.254
Precision	94.102
Recall	90.314

	Predicted: Computer	Predicted: Recreational
Actual: Computer	1470	90
Actual: Recreational	154	1436



We can see that LR and SVM have nearly the same area under the ROC, implying that they classify records most accurately, while Naive Bayes lacks the same accuracy.

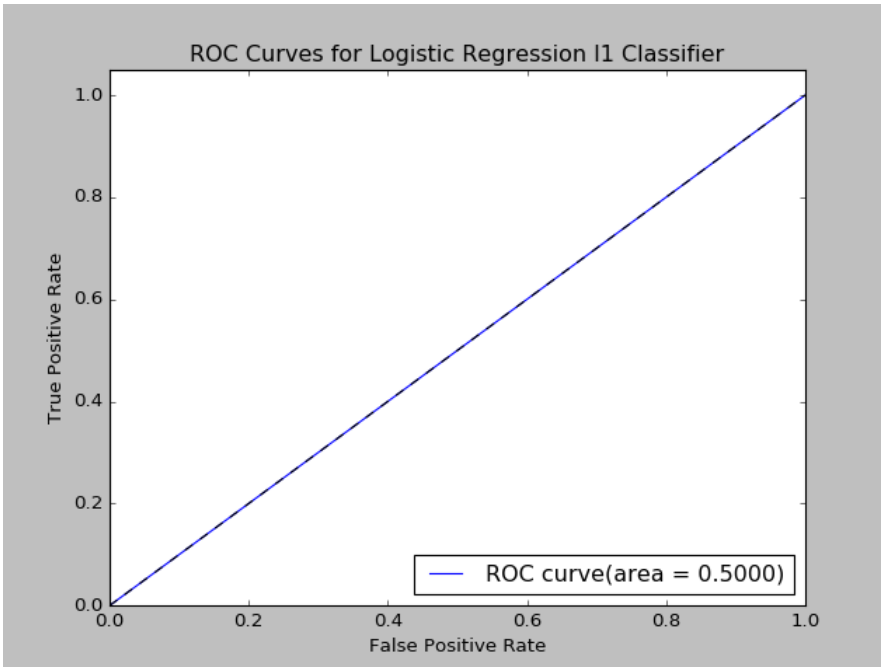
5.5 Question (i) - Logistic Regression with Regularization term

Now, we repeat the tests performed in question (h), but we've now added a regularization term to the optimization objective. Both l1 and l2 norm regularizations, varying the regularization parameter from 0.1 to 1000.

5.5.1 $k = 10^{-3} \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	50.407
Precision	00.000
Recall	00.000

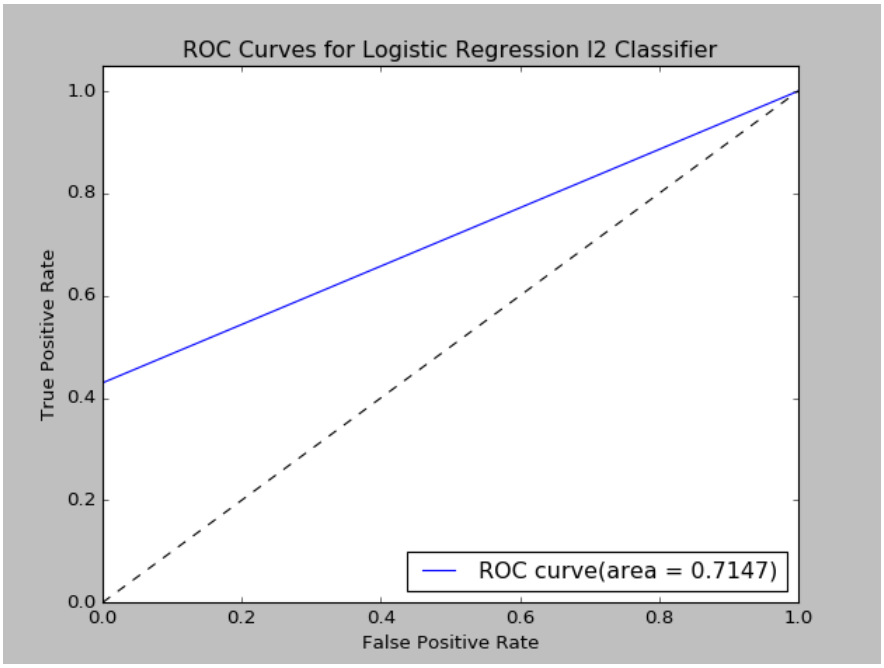
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1590	0
Actual: Recreational	1560	0



5.5.2 $k = 10^{-3} \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	71.746
Precision	100.000
Recall	42.949

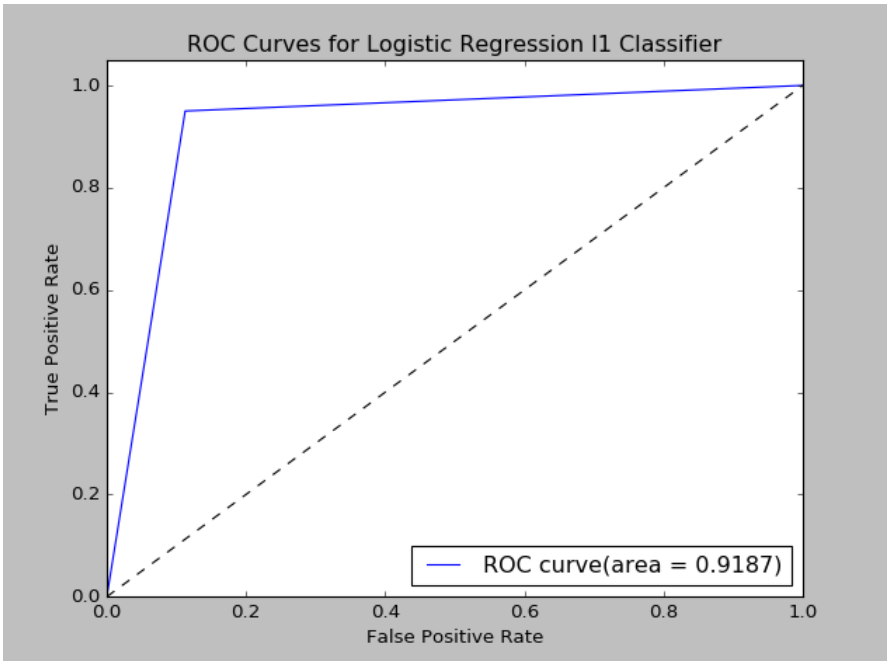
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1590	0
Actual: Recreational	890	670



5.5.3 $k = 10^{-2} \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	91.841
Precision	89.223
Recall	95.000

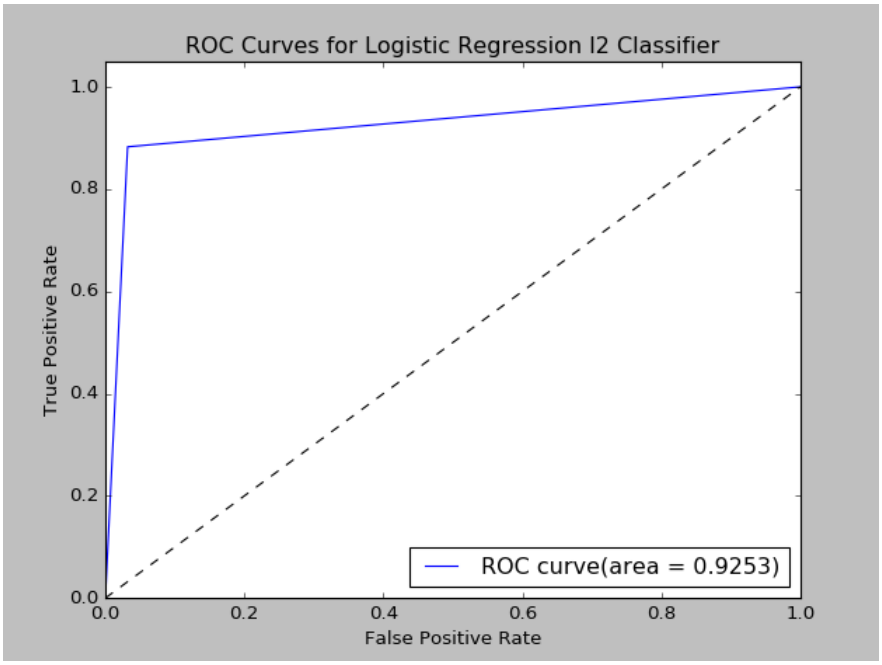
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1411	179
Actual: Recreational	78	1482



5.5.4 $k = 10^{-2} \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	92.571
Precision	96.428
Recall	88.270

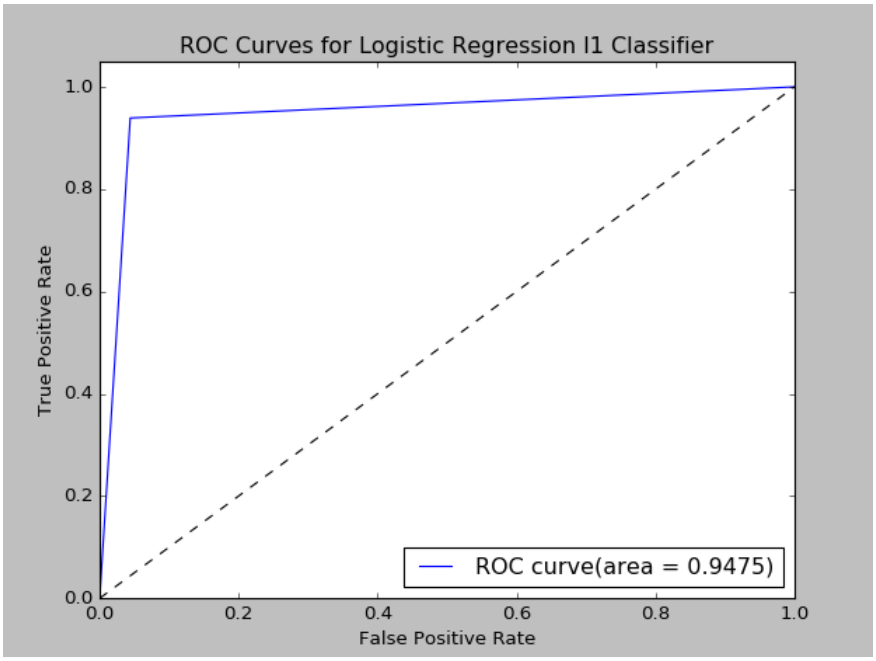
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1539	51
Actual: Recreational	183	1377



5.5.5 $k = 10^{-1} \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	94.762
Precision	95.440
Recall	93.910

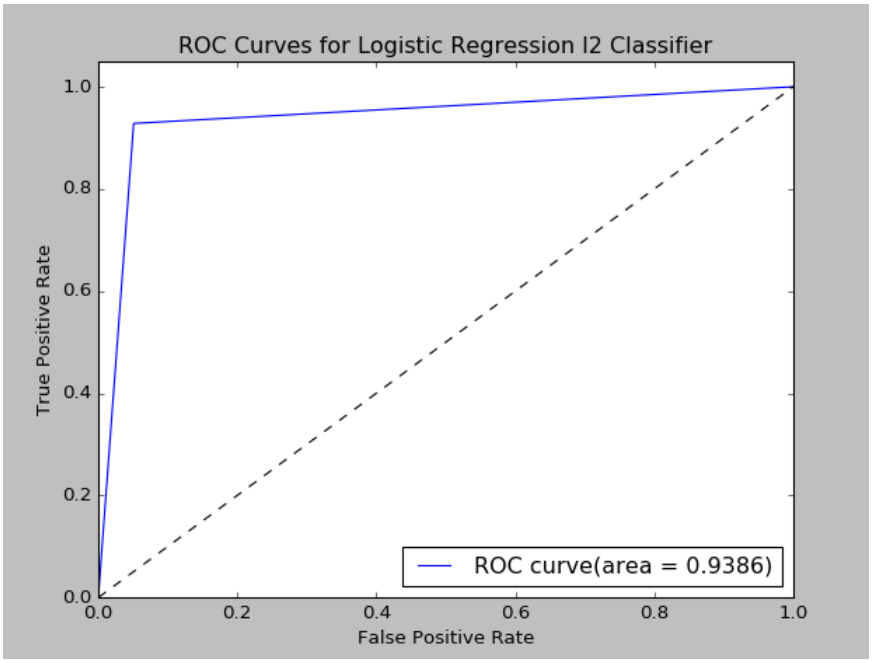
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1520	70
Actual: Recreational	95	1465



5.5.6 $k = 10^{-1} \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	93.873
Precision	94.702
Recall	92.820

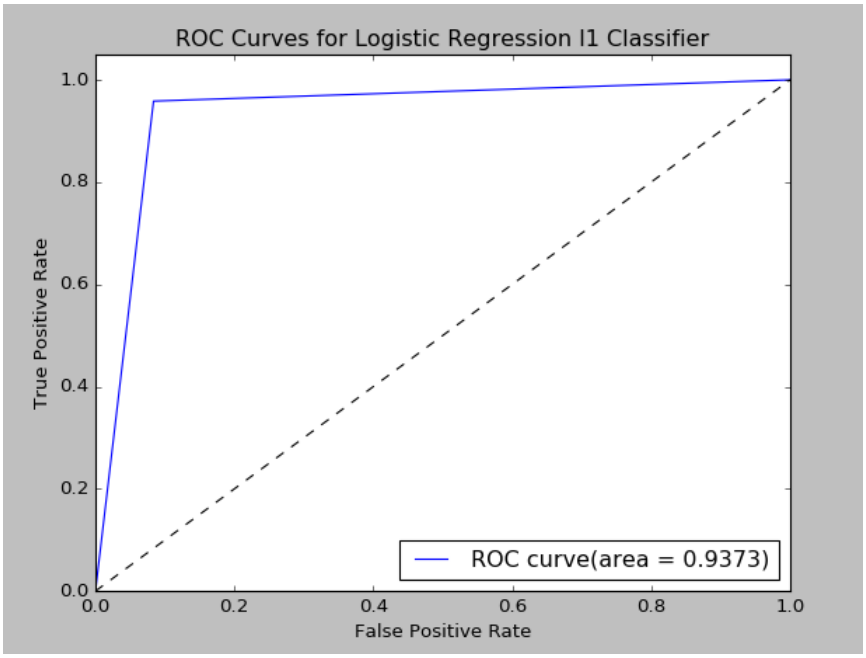
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1509	81
Actual: Recreational	112	1448



5.5.7 $k = 10^0 \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	93.714
Precision	91.830
Recall	95.833

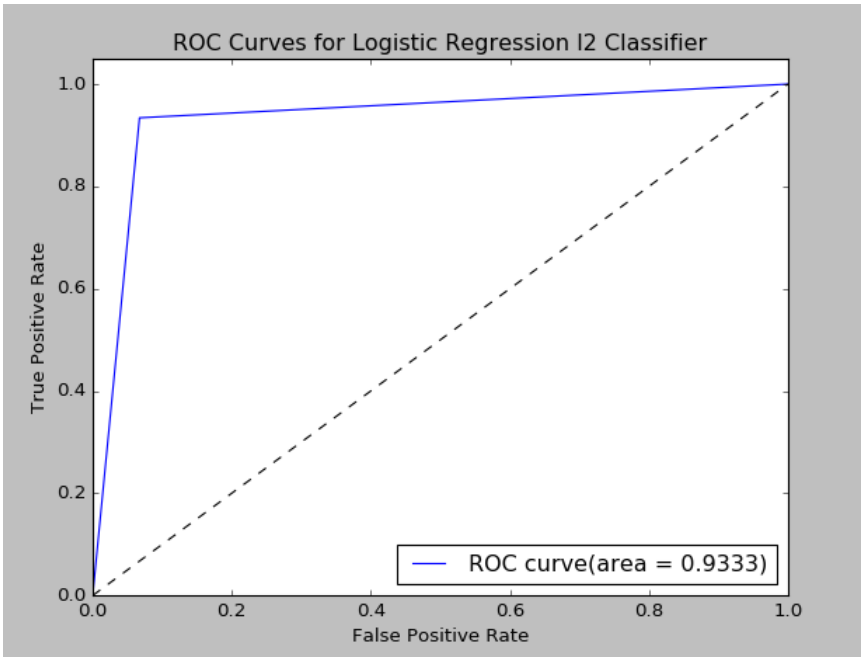
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1457	133
Actual: Recreational	65	1495



5.5.8 $k = 10^0 \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	93.333
Precision	93.158
Recall	93.397

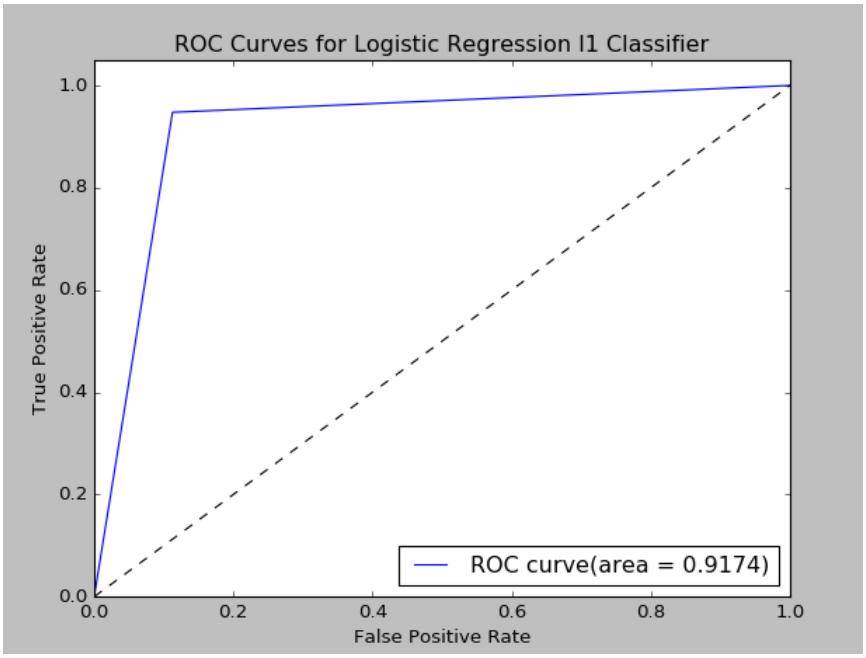
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1483	107
Actual: Recreational	103	1457



5.5.9 $k = 10^1 \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	91.714
Precision	89.197
Recall	94.743

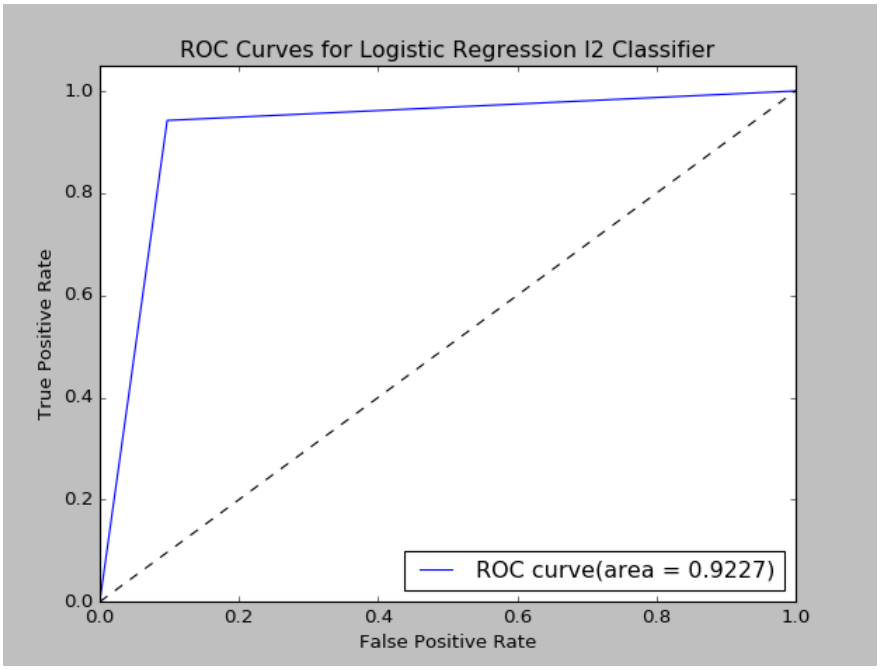
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1411	179
Actual: Recreational	82	1478



5.5.10 $k = 10^1 \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	92.254
Precision	90.517
Recall	94.231

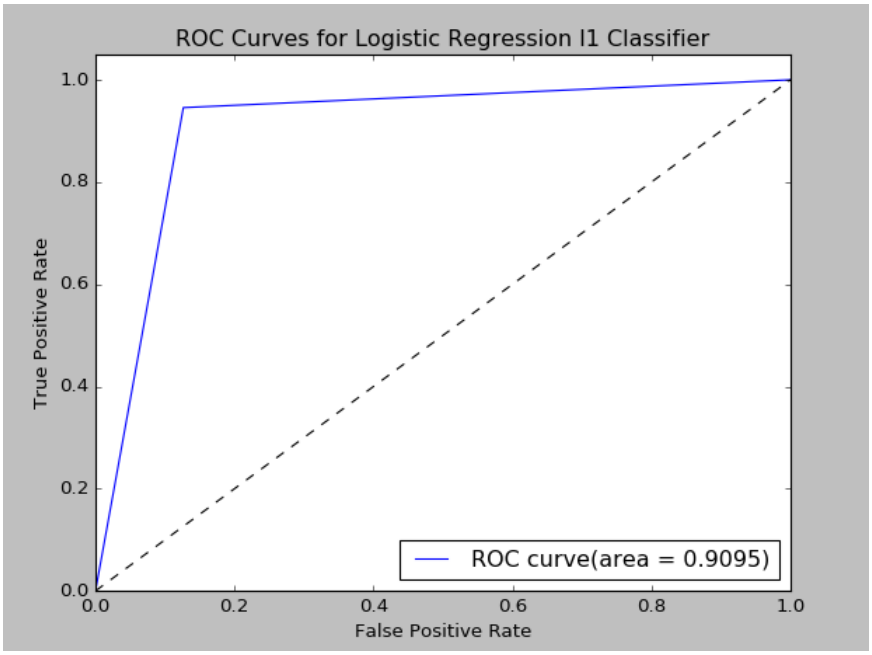
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1436	154
Actual: Recreational	90	1470



5.5.11 $k = 10^2 \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	90.920
Precision	88.007
Recall	94.551

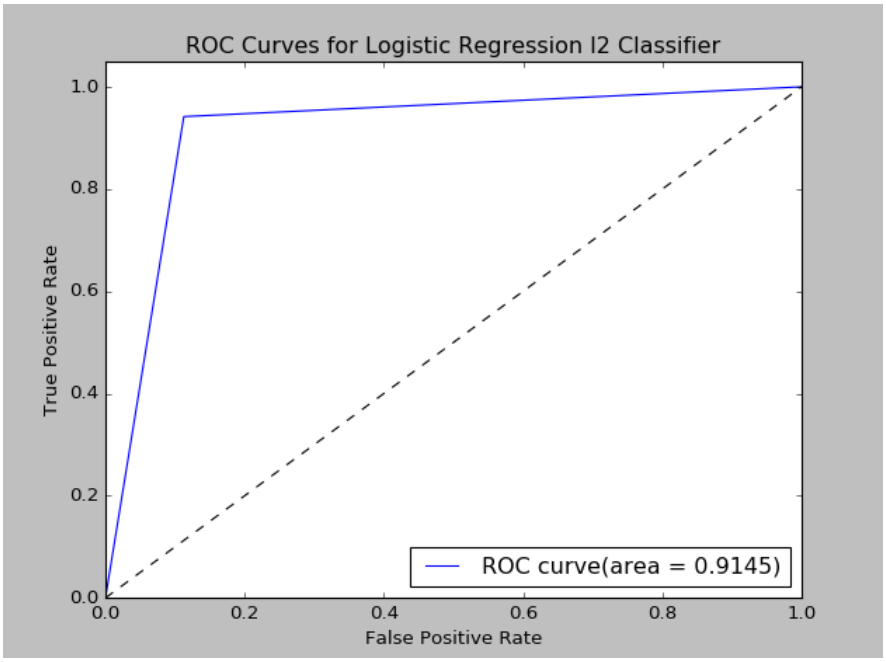
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1389	201
Actual: Recreational	85	1475



5.5.12 $k = 10^2 \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	91.428
Precision	89.138
Recall	94.167

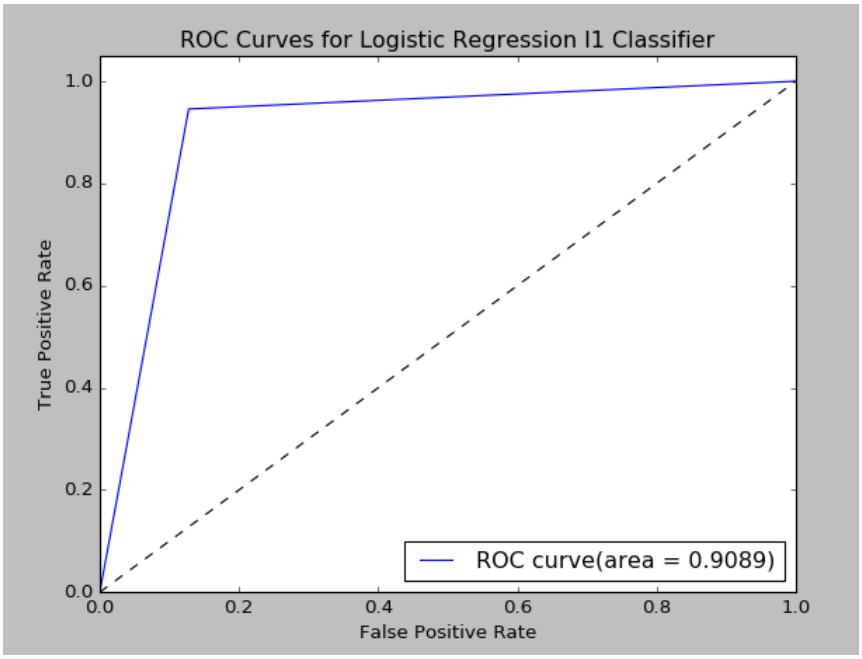
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1411	179
Actual: Recreational	91	1469



5.5.13 $k = 10^3 \rightarrow l_1$ *regularization*

Statistic	Result
Accuracy	90.857
Precision	87.902
Recall	94.551

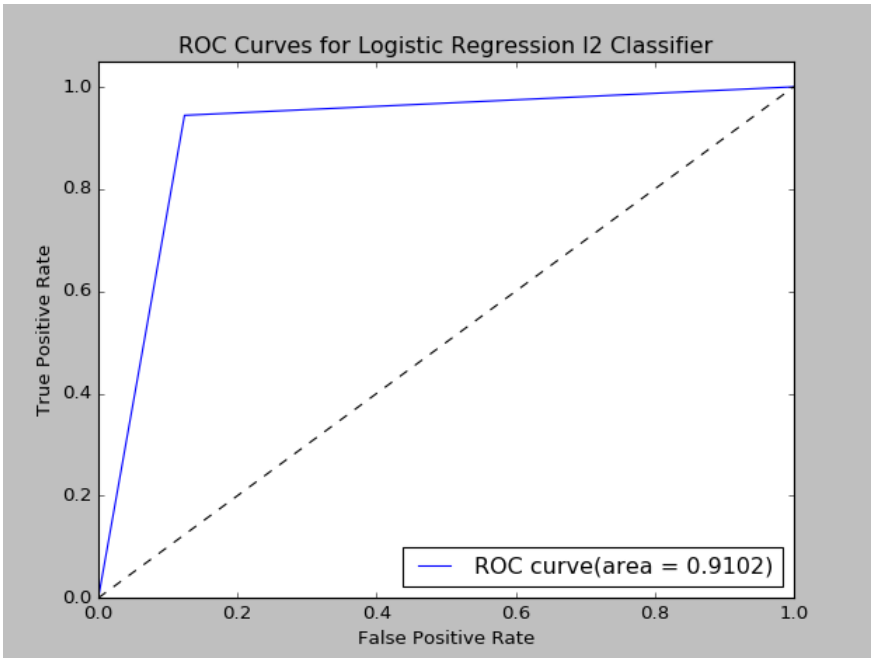
	Predicted: Computer	Predicted: Recreational
Actual: Computer	1387	203
Actual: Recreational	85	1475



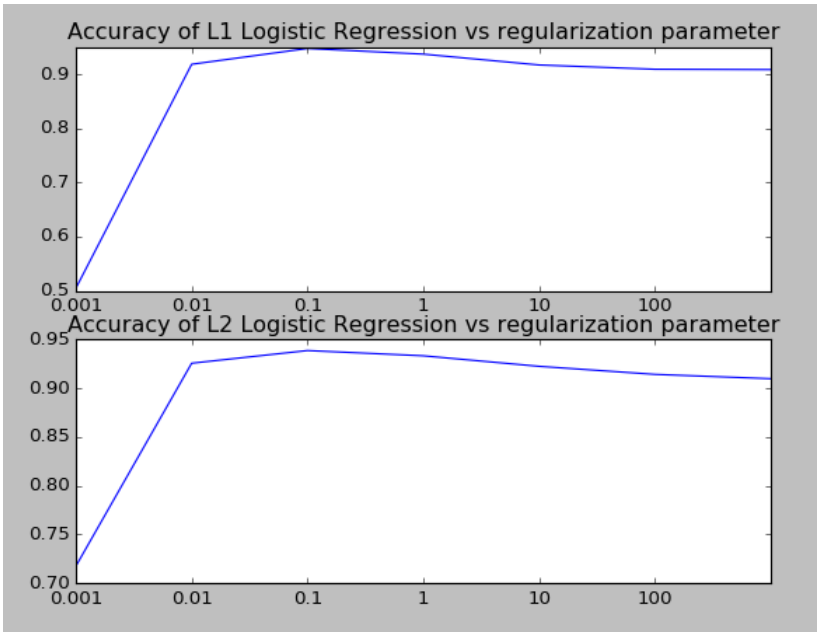
5.5.14 $k = 10^3 \rightarrow l_2$ *regularization*

Statistic	Result
Accuracy	90.984
Precision	88.203
Recall	94.423

	Predicted: Computer	Predicted: Recreational
Actual: Computer	1393	197
Actual: Recreational	87	1473



5.5.15 Final comparison



Testing error vs regularization parameter

Parameter	L1	L2
-3	49.593	28.254
-2	8.159	7.429
-1	5.238	6.127
0	6.286	6.667
1	8.286	7.746
2	9.08	8.872
3	9.143	9.016

Coefficient vs regularization parameter

Parameter	L1	L2
-3	0.000	-0.002
-2	-0.127	-0.034
-1	-1.203	-0.333
0	-1.915	-0.726
1	-1.408	-1.395
2	-1.742	-1.402
3	-1.905	-1.501

We would typically use L1 loss function when we need a robust solution, but have the computational power, and could to tolerate multiple stable solutions. If the dataset is very large, or if a single not stable solution will work, we would use the L2 loss function.

We also noticed that as the regularization parameter increased, the fitted hyperplane moves away from the origin.

6 Multiclass Classification

6.1 Question (i) - Multiclass Classification - Naive Bayes and SVM

We train classifiers on the documents belonging to the classes A (comp.sys.ibm.pc.hardware), B (comp.sys.mac.hardware), C (misc.forsale), and D (soc.religion.christian). We use SVM and Naive Bayes in One Vs One and One Vs Rest methods to classify the dataset. The classifier statistics are as follows:

6.1.1 OneVsOne - Naive Bayes

Statistic	Result
Accuracy	63.130
Precision	64.049
Recall	62.973

	Predicted: B	Predicted: B	Predicted: C	Predicted: D
Actual: A	199	94	94	5
Actual: B	101	189	94	1
Actual: C	66	74	249	1
Actual: D	1	19	27	351

6.1.2 OneVsOne - SVM

Statistic	Result
Accuracy	73.866
Precision	74.287
Recall	73.782

	Predicted: B	Predicted: B	Predicted: C	Predicted: D
Actual: A	198	155	39	0
Actual: B	99	263	23	0
Actual: C	43	20	327	0
Actual: D	8	10	12	368

6.1.3 OneVsRest - Naive Bayes

Statistic	Result
Accuracy	62.620
Precision	63.162
Recall	62.462

	Predicted: B	Predicted: B	Predicted: C	Predicted: D
Actual: A	181	97	102	12
Actual: B	80	191	107	7
Actual: C	52	80	252	6
Actual: D	0	21	21	356

6.1.4 OneVsRest - SVM

Statistic	Result
Accuracy	73.930
Precision	73.775
Recall	73.838

	Predicted: B	Predicted: B	Predicted: C	Predicted: D
Actual: A	174	162	55	1
Actual: B	89	263	33	0
Actual: C	25	24	339	2
Actual: D	0	11	6	381