EE 219 Project II - Report

Muralidharan, Vignesh UID: 904729596
Muthappan, Chidambaram UID: 704774938
Jeyakumar, Jeya Vikranth UID: 404749568

February 15, 2017

Abstract

The purpose of this project is to perform classification analysis on the '20 Newsgroups dataset' to group them into two classes: Computer Technology and Recreational activity and also to understand the purpose of model evaluation and the common evaluation procedures. The classification accuracy of different models is calculated and the limitations of each model is studied. Confusion matrix is found to describe the performance of the classifier and various metrics like precision, recall etc. are calculated from the confusion matrix. Then the ROC curve is plotted and the Area under the curve(AUC) is measured to further understand the different models.

1 Dataset:

1.1 Question a

The 20 Newsgroup dataset is imported and the data from the following eight categories are loaded.

- 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

Now a histogram of number of documents per topic is plotted and we can see from the figure that the number of documents per topic is evenly distributed. Also the total number of documents in the two groups (Computer Technology and Recreational Activity) is plotted.

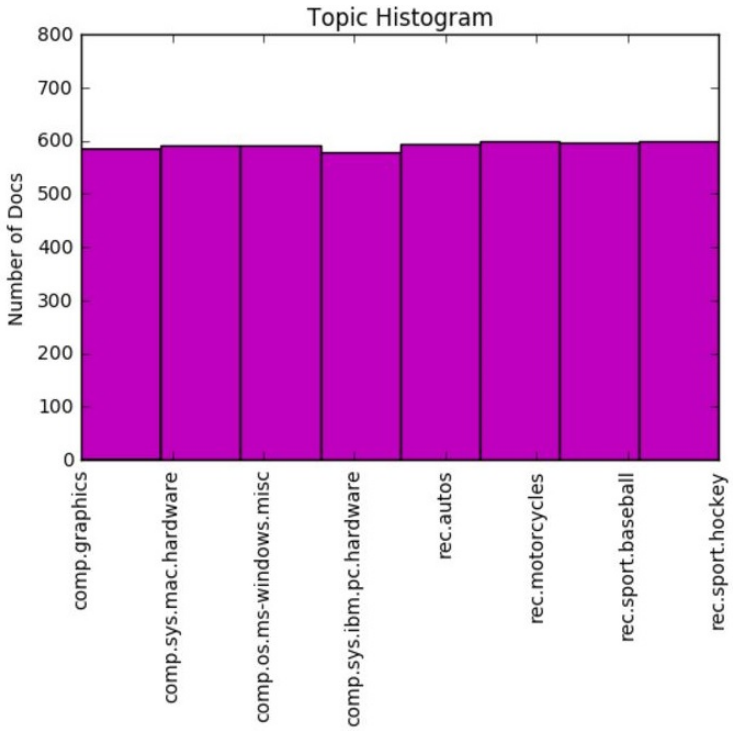

Figure 1: Number of documents in each Topic

```
Number of Documents :-
 in Computer Technology: 2343
 in Recreational Activity: 2389
```

2 Modeling Text Data and Feature Extraction:

2.1 Question b

From the scikit-learn documentation:
Text Analysis is a major application field for machine learning algorithms. However the raw data, a sequence of symbols cannot be fed directly to the algorithms themselves as most of them expect numerical feature vectors with a fixed size rather than the raw text documents with variable length. We will use CountVectorizer to "convert text into a matrix of token counts"

Features and samples are defined as follows:

- Each individual token occurrence frequency (normalized or not) is treated as a feature.
- The vector of all the token frequencies for a given document is considered a multivariate sample..

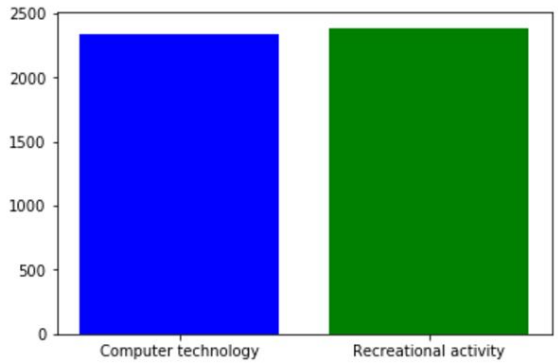

Figure 2: Topic: Number of documents in Each Group

A corpus of documents can thus be represented by a matrix with one row per document and one column per token (e.g. word) occurring in the corpus.

We call vectorization the general process of turning a collection of text documents into numerical feature vectors. This specific strategy (tokenization, counting and normalization) is called the Bag of Words or "Bag of n-grams" representation. Documents are described by word occurrences while completely ignoring the relative position information of the words in the document.

We defined our tokenizer such that has only letters by using Regular expressions filter and also eliminated the characters with Ascii values greater than 128 by using the ord function. After removing all the punctuations and symbols we did stemming to remove the words with the same meaning by using the SnowBall Stemmer. All the common words were also removed from the vocabulary by declaring them as stop words. Terms that occur in less than four documents(very rarely occurring words) were also removed by setting the parameter `min_df = 4`

Then we did the TF-IDF transformation to get the TF-IDF vector representation. TF-IDF stands for "Term Frequency, Inverse Document Frequency". It is a way to score the importance of words (or "terms") in a document based on how frequently they appear across multiple documents.

```
Dimensions of Numerical feature vector: (4732, 10743)
Dimensions of TF-IDF vector: (4732, 10743)
Number of terms Extracted: 10743
```

2.2 Question c

For this Question the data from all 20 Newsgroups were loaded and then TFxICF was calculated by using the formula

$$TF \times ICF = (0.5 + 0.5\frac{f_{t,c}}{max_{f_{t',c}}}) \times log\left[\frac{|C|}{|c \epsilon C : t \epsilon c|'}\right] \quad (1)$$

10 most significant terms in the following classes, with respect to the above measure is calculated.
comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, misc.forsale, and soc.religion.christian.

```
Most significant 10 terms in comp.sys.ibm.pc.hardware
['scsihasi', 'buslog', 'michaeljestergund', 'megssec', 'latonia',
'mbyet', 'fasst', 'dossi', 'aspi', 'schaufenbuel']

Most significant 10 terms in comp.sys.mac.hardware
['firstclass', 'powerbook', 'hadescoosdartmouthedu', 'zterm',
'bmug', 'techwork', 'iivx', 'lciii', 'iisi', 'iifx']

Most significant 10 terms in misc.forsale
['snes', 'sabretooth', 'kou', 'keown', 'liefeld',
'xforc', 'uccxkvb', 'tnde', 'spiderman', 'hobgoblin']

Most significant 10 terms in soc.religion.christian
['monophysit', 'jaynemmaltguildorg', 'kulikauska', 'sspx',
'jayn', 'athosrutgersedu', 'liturgi', 'genevarutgersedu',
'schismat', 'atterlepvelaacsoaklandedu']
```

3 Feature Selection:

3.1 Question d

The TF-IDF vectors is in the order of thousands and are very sparse, i.e, most of their entries is zero. So in order to improve the performance of the machine learning algorithms these vectors are converted to low dimensions by using the transform called Latent Semantic Indexing (LSI) In our project we used the Truncated SVD function to perform LSI and reduce the dimensionality of the matrix to 50 by setting the rank parameter k=50. But the LSI transformation may lead to values less than zero and these values cannot be used directly in training algorithms. So we use the MinMaxScalar function to map the LSI values between 0 and 1. Next we use this reduced dimensional matrix in our different learning algorithms for the following questions.

```
Dimensions of TF-IDF vector after LSI: (4732, 50)
```

4 Learning Algorithms:

In our Project we have studied and analyzed the following learning algorithms for classification.

- Linear Support Vector Machines
- Soft Margin SVM
- Naïve Bayes Algorithm (Multinomial)

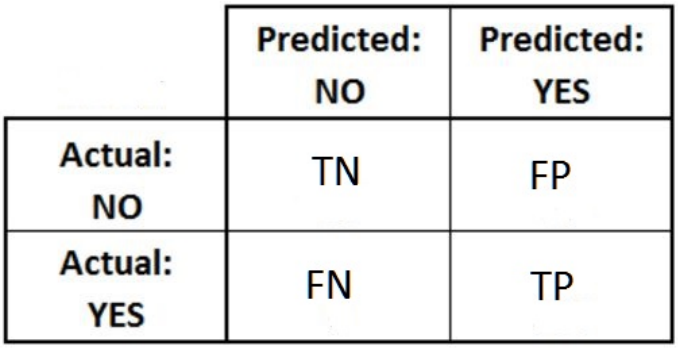

Figure 3: Confusion Matrix

- Logistic Regression Classifier
- Logistic Regression Classifier with Regularization

Before we start looking at the results for these different algorithms let us first understand the various Model Evaluation Metrics based on which the algorithms are analyzed.

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 Confusion matrix gives you a more complete picture of how your classifier is performing Also allows you to compute various classification metrics, and these metrics can guide your model selection
Basic terminologies:

- 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

Classification accuracy: percentage of correct predictions
Classification accuracy is the easiest classification metric to understand but, it does not tell us the underlying distribution of response values. It also does not tell us what "types" of errors our classifier is making.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{2}$$

Recall/True Positive Rate: Tells us how often the prediction is correct When the actual value is positive.Also known as "True Positive Rate" or "Sensitivity"

$$Recall = TP/(TP + FN) \tag{3}$$

Precision: Tells us how often the prediction is correct when a positive value is predicted.

$$Precision = TP/(TP + FP) \tag{4}$$

False Positive Rate: Tells us how often the prediction is incorrect when the actual value is negative.

$$FPR = FP/(TN + FP) \tag{5}$$

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.

Confusion Matrix Advantages:

- Allows you to calculate a variety of metrics
- Useful for multi-class problems (more than two response classes)

ROC/AUC Advantages:

- Does not require you to set a classification threshold
- Still useful when there is high class imbalance

4.1 Question e - Linear Support Vector Machines

In machine learning, support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

Basically, SVM model is a representation of the examples as points in space which are mapped such that the examples of the separate categories are divided by a clear gap that is as wide as possible. Thereafter, new examples are then mapped into that same space and predicted to belong to a particular category based on which side of the gap they fall.

The algorithmic steps involved in this process are:

- First obtain test data and perform above transformations.

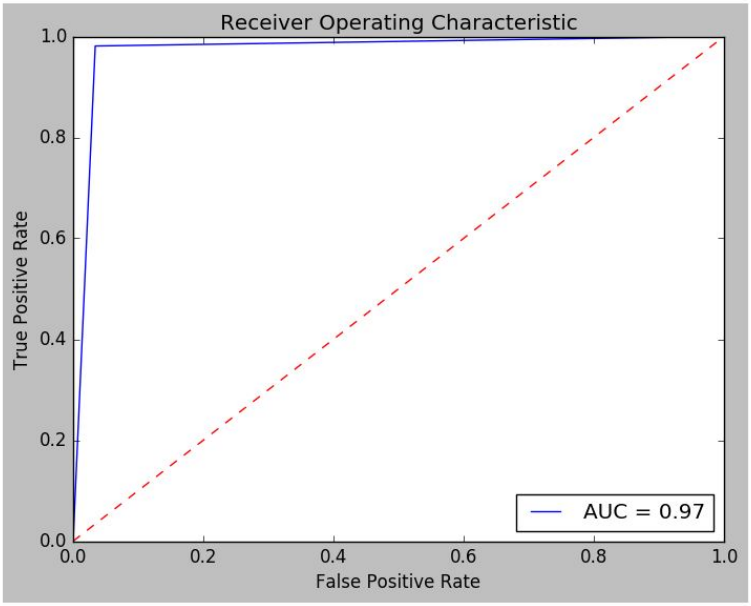

Figure 4: ROC and AUC of Linear SVM

- Second reduce multiclass problem to binary classification problem by reducing all subclasses of comp and subclasses of rec for train and for test data, classify it as, comp = 0, rec = 1.
- Train model and predict labels for test set , ie., measure accuracy between linear and predicted SVM outputs.
- Report the results, for test targets and predicted SVMs.

```
==========SVM==========
                        precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.973650793651
Confusion Matrix
[[1507   53]
 [  30 1560]]
```

4.2 Question f - Soft Margin SVM

For the very high dimensional problems common in text classification, sometimes the data are linearly separable. But in the general case they are not, and

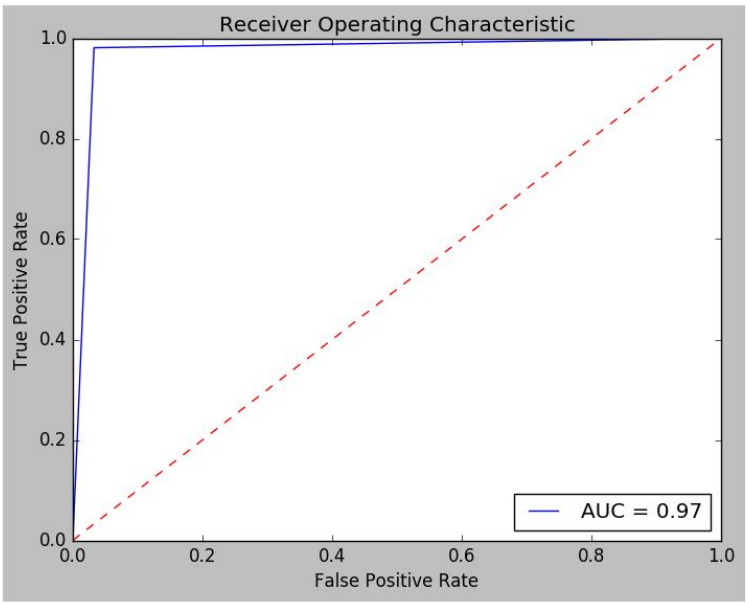

Figure 5: ROC and AUC of Soft Margin SVM

even if they are, we might prefer a solution that better separates the bulk of the data while ignoring a few weird noise documents.

The allowance of softness in margins (i.e. a low cost setting) allows for errors to be made while fitting the model (support vectors) to the training/discovery data set. Conversely, hard margins will result in fitting of a model that allows zero errors. Sometimes it can be helpful to allow for errors in the training set, because it may produce a more generalizable model when applied to new datasets. Forcing rigid margins can result in a model that performs perfectly in the training set, but is possibly over-fit / less generalizable when applied to a new dataset. Identifying the best settings for 'cost' is probably related to the specific data set you are working with.

```
Best Gamma Value: 1000
==========soft_margin_SVM==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.974285714286
Confusion Matrix
[[1509   51]
 [  30 1560]]
```

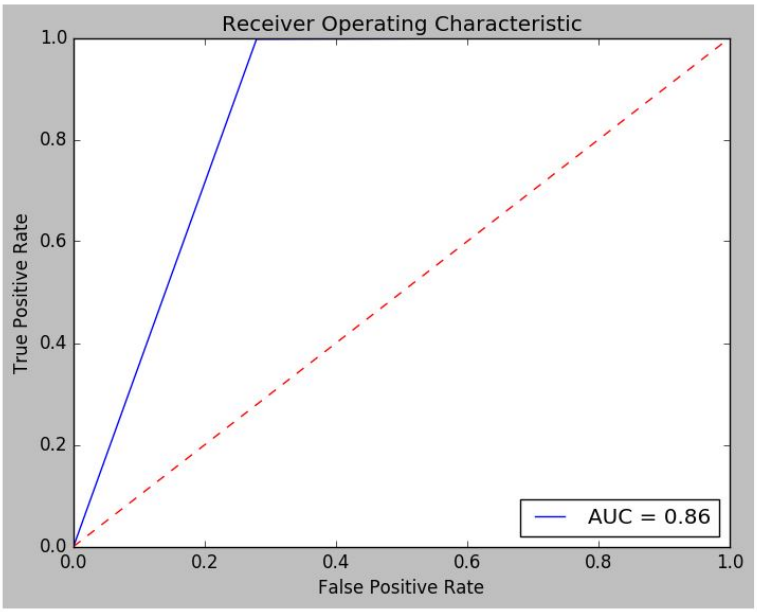

Figure 6: ROC and AUC of Naive Bayes Algorithm

4.3 Question g - Naïve Bayes Algorithm

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

```
===========Naive Bayes===========
                       precision    recall  f1-score   support

  Computer technology       1.00      0.72      0.84      1560
Recreational activity       0.78      1.00      0.88      1590

          avg / total       0.89      0.86      0.86      3150

accuracy: 0.860634920635
Confusion Matrix
[[1125  435]
 [   4 1586]]
```

4.4 Question h - Logistic Regression Classifier

Logistic regression, despite its name, is a linear model for classification rather than regression. Logistic regression is also known in the literature as logit regression, maximum-entropy classification (MaxEnt) or the log-linear classifier. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.

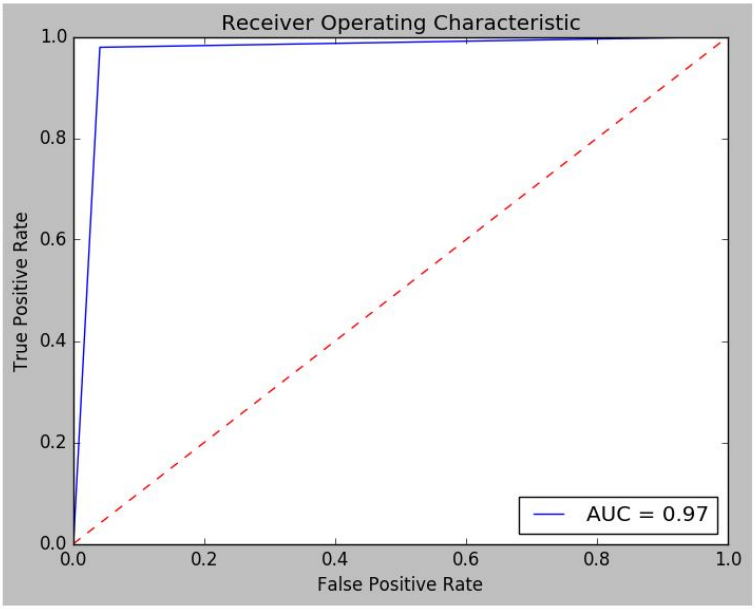

Figure 7: ROC and AUC of Logistic Regression

```
===========Logistic Regularization(Unregularized)===========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.96      0.97      1560
Recreational activity       0.96      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.969523809524
Confusion Matrix
[[1497   63]
 [  33 1557]]
```

4.5 Question i - Regularized Logistic Regression Classifier

Sometimes overfitting occurs with Logistic Regression when trying to fit a high order polynomial function. One of the ways to deal with overfitting is Regularization. Regularization keeps all the features, but reduces the magnitude of parameters θ Works well when we have a lot of features, each of which contributes a bit to predicting y

λ is the regularization parameter and it controls a trade off between our two goals

- To fit the training set well
- To keep parameters small

L2 regularization	L1 regularization
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases
Non-sparse outputs	Sparse outputs
No feature selection	Built-in feature selection

Figure 8: L1 Regularization vs L2 Regularization

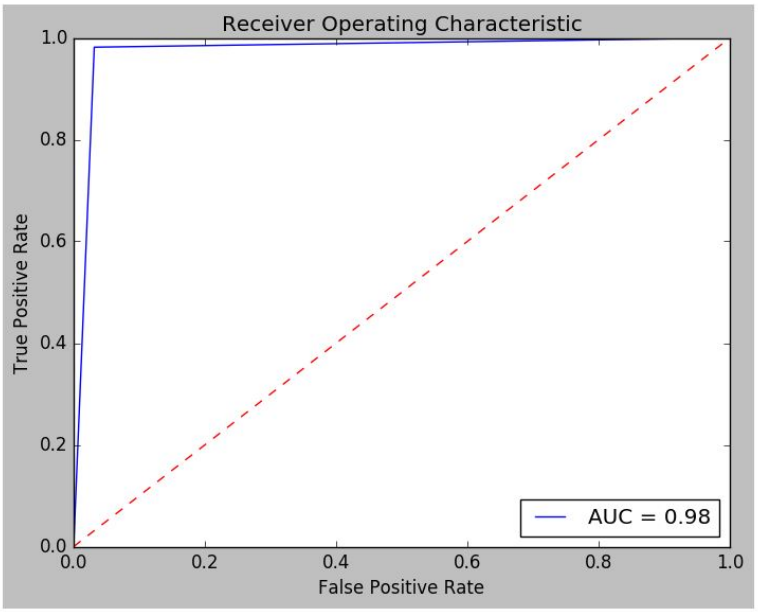

Figure 9: ROC and AUC of Logistic Regression: L1 , lambda = 0.0001

Using the regularized objective (i.e. the cost function with the regularization term) we get a much smoother curve which fits the data and gives a much better hypothesis If λ is very large we end up penalizing ALL the parameters and so all the parameters end up being close to zero If this happens, it's like we got rid of all the terms in the hypothesis and this results in underfitting So the hypothesis would be too biased because of the absence of any parameters (effectively) So,λ should be chosen carefully and should not be too big.

L1 penalty gives us sparser solutions than L2 penalty. L1 regularization helps perform feature selection in sparse feature spaces, and that is a good practical reason to use L1 in some situations. But usually L2 regularization gives us a better solution than L1 regularization.

The following are the results for **L1 Regularization**.

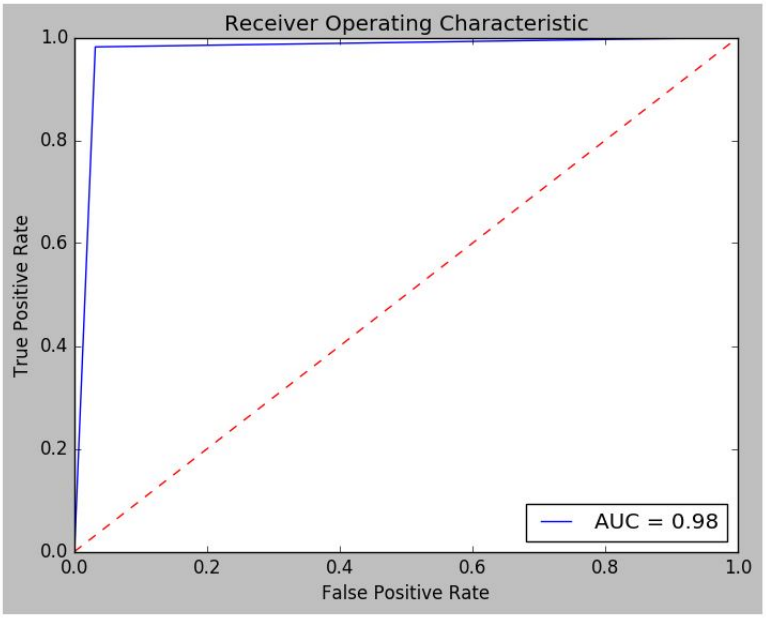

Figure 10: ROC and AUC of Logistic Regression: L1 , lambda = 0.001

```
==========Logistic Regression: L1  , lambda = 0.0001==========
                     precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.98      1590

          avg / total       0.98      0.98      0.98      3150

accuracy: 0.975238095238
Confusion Matrix
[[1511   49]
 [  29 1561]]
```

```
==========Logistic Regression: L1 , lambda = 0.001==========
                     precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.98      1590

          avg / total       0.98      0.98      0.98      3150

accuracy: 0.975238095238
Confusion Matrix
[[1511   49]
 [  29 1561]]
```

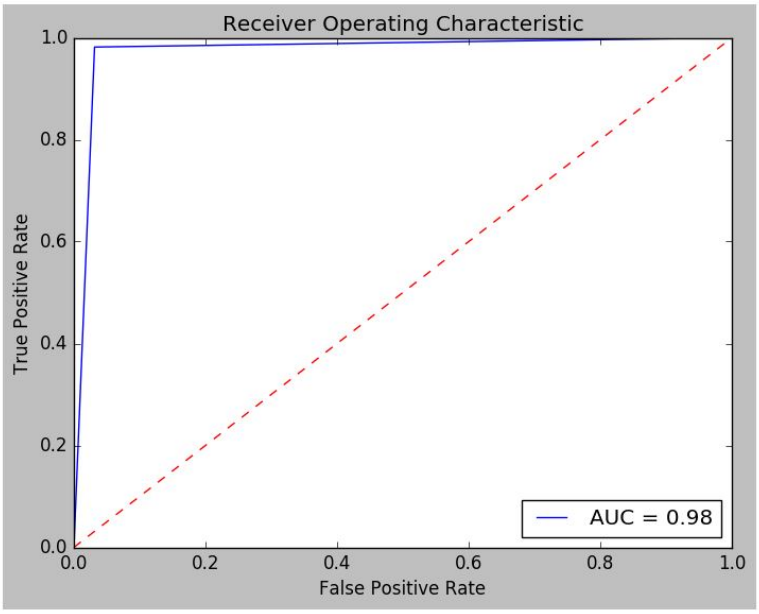

Figure 11: ROC and AUC of Logistic Regression: L1 , lambda = 0.01

```
==========Logistic Regression: L1 , lambda = 0.01==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.98      1590

          avg / total       0.98      0.98      0.98      3150

accuracy: 0.975238095238
Confusion Matrix
[[1511   49]
 [  29 1561]]
```

```
==========Logistic Regression: L1 , lambda = 0.5==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.973333333333
Confusion Matrix
[[1507   53]
 [  31 1559]]
```

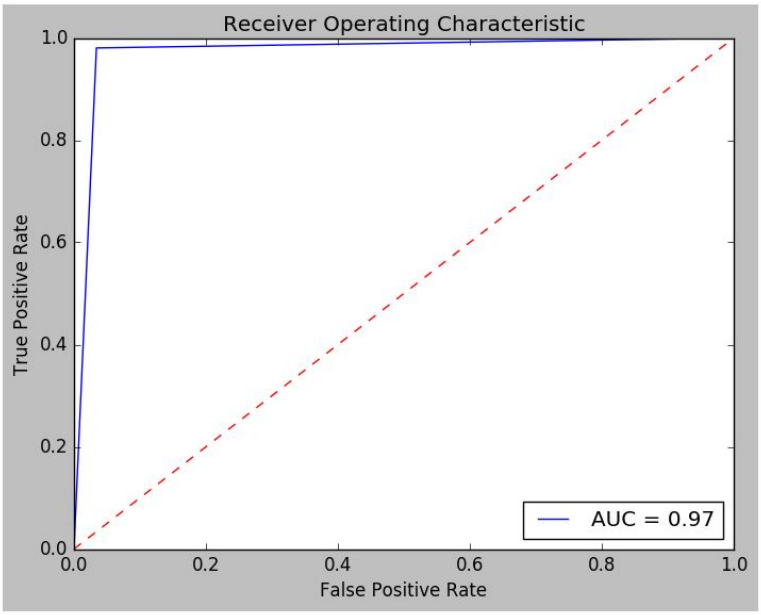

Figure 12: ROC and AUC of Logistic Regression: L1 , lambda = 0.5

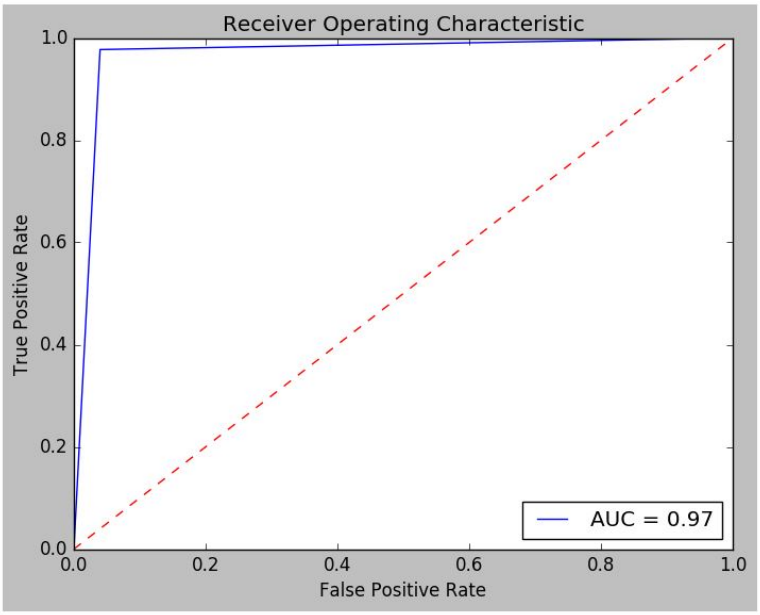

Figure 13: ROC and AUC of Logistic Regression: L1 , lambda = 1.0

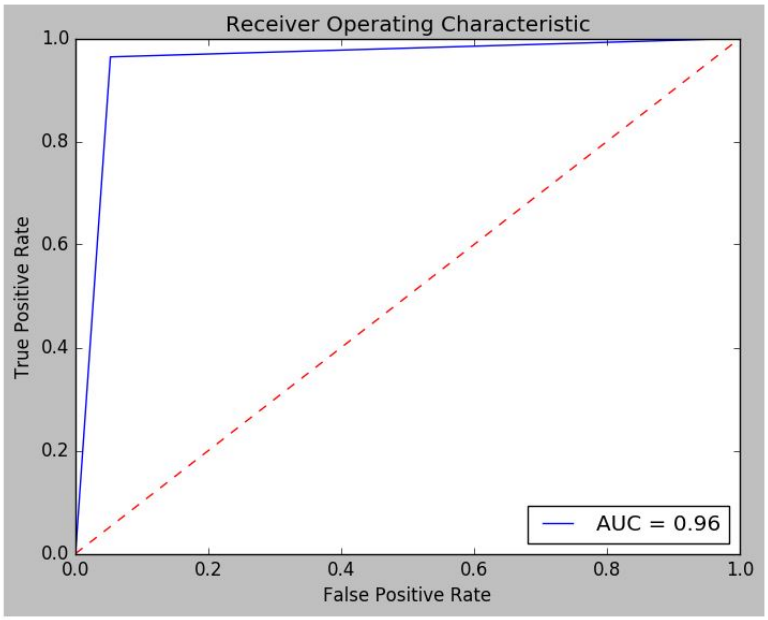

Figure 14: ROC and AUC of Logistic Regression: L1 , lambda = 10.0

```
==========Logistic Regression: L1 , lambda = 1.0==========
                     precision    recall  f1-score   support

  Computer technology       0.98      0.96      0.97      1560
Recreational activity       0.96      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.968888888889
Confusion Matrix
[[1498   62]
 [  36 1554]]
```

```
==========Logistic Regression: L1 , lambda = 10.0==========
                     precision    recall  f1-score   support

  Computer technology       0.96      0.95      0.96      1560
Recreational activity       0.95      0.96      0.96      1590

          avg / total       0.96      0.96      0.96      3150

accuracy: 0.955873015873
Confusion Matrix
[[1478   82]
 [  57 1533]]
```

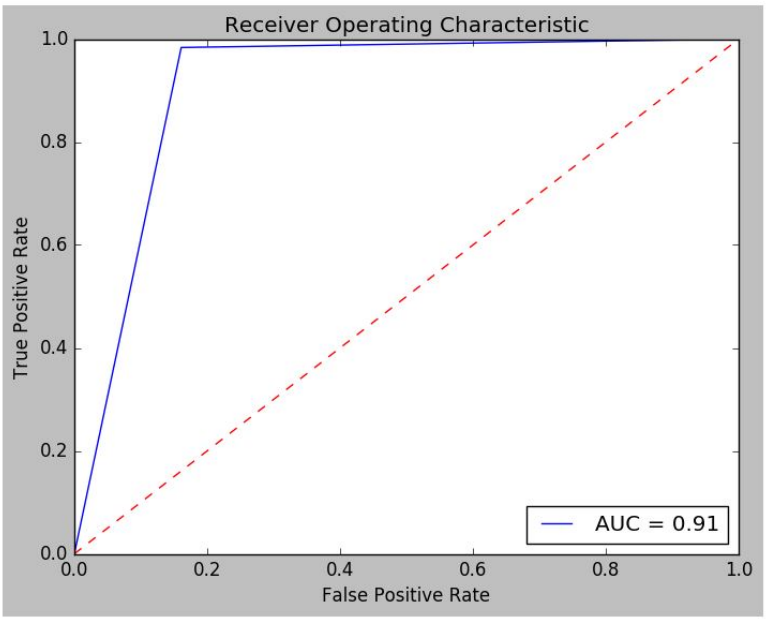

Figure 15: ROC and AUC of Logistic Regression: L1 , lambda = 100.0

```
==========Logistic Regression: L1 , lambda = 100.0==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.84      0.90      1560
Recreational activity       0.86      0.98      0.92      1590

          avg / total       0.92      0.91      0.91      3150

accuracy: 0.912063492063
Confusion Matrix
[[1309  251]
 [  26 1564]]
```

The following are the results for **L2 Regularization**.

```
==========Logistic Regression: L2 , lambda = 0.0001==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.98      1590

          avg / total       0.98      0.98      0.98      3150

accuracy: 0.975238095238
Confusion Matrix
```

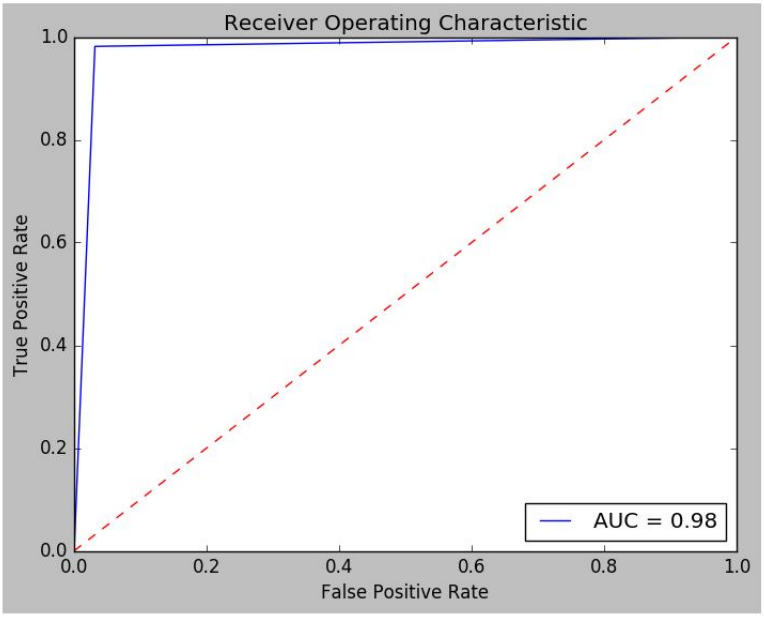

Figure 16: ROC and AUC of Logistic Regression: L2 , lambda = 0.0001

```
[[1511   49]
 [  29 1561]]
```

```
==========Logistic Regression: L2 , lambda = 0.001==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.98      1590

          avg / total       0.98      0.97      0.97      3150

accuracy: 0.974920634921
Confusion Matrix
[[1510   50]
 [  29 1561]]
```

```
==========Logistic Regression: L2 , lambda = 0.01==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.97      0.97      1560
Recreational activity       0.97      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.973968253968
```

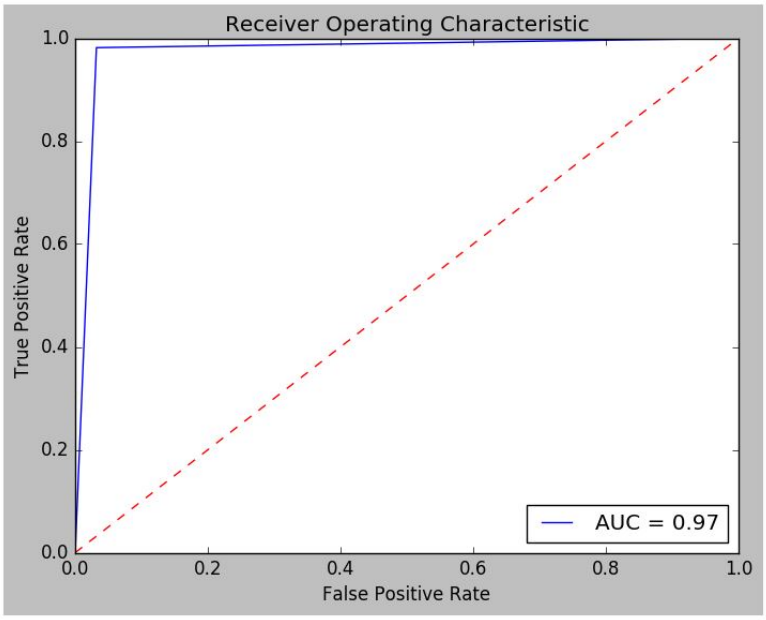

Figure 17: ROC and AUC of Logistic Regression: L2 , lambda = 0.001

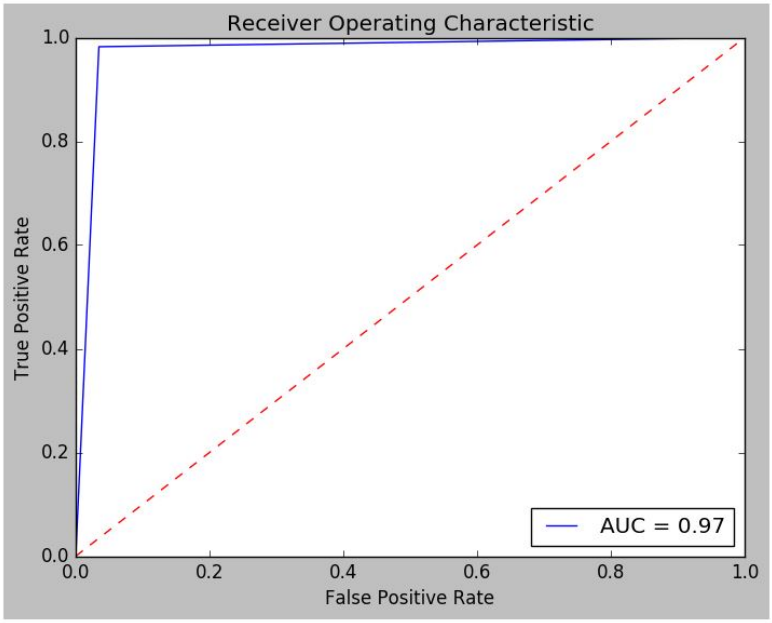

Figure 18: ROC and AUC of Logistic Regression: L2 , lambda = 0.01

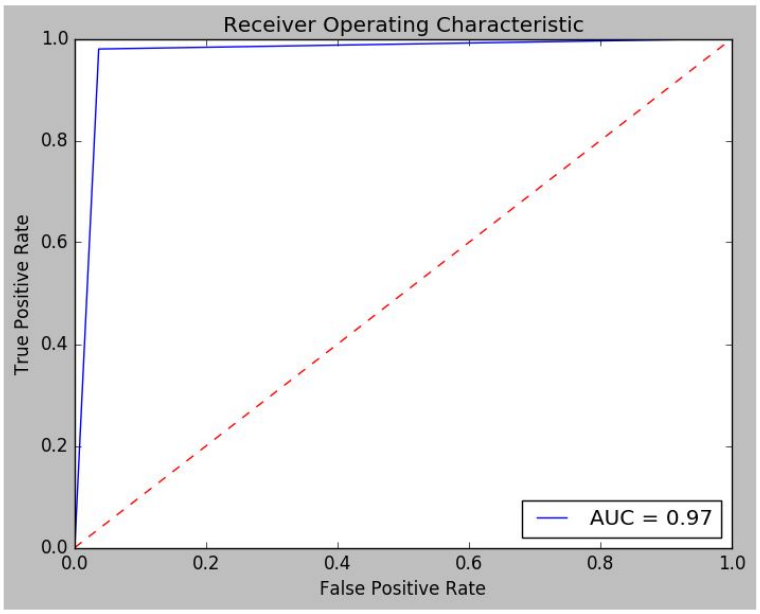

Figure 19: ROC and AUC of Logistic Regression: L2 , lambda = 0.5

```
Confusion Matrix
[[1506   54]
 [  28 1562]]
```

```
==========Logistic Regression: L2 , lambda = 0.5==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.96      0.97      1560
Recreational activity       0.96      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150

accuracy: 0.971746031746
Confusion Matrix
[[1503   57]
 [  32 1558]]
```

```
==========Logistic Regression: L2 , lambda = 1.0==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.96      0.97      1560
Recreational activity       0.96      0.98      0.97      1590

          avg / total       0.97      0.97      0.97      3150
```

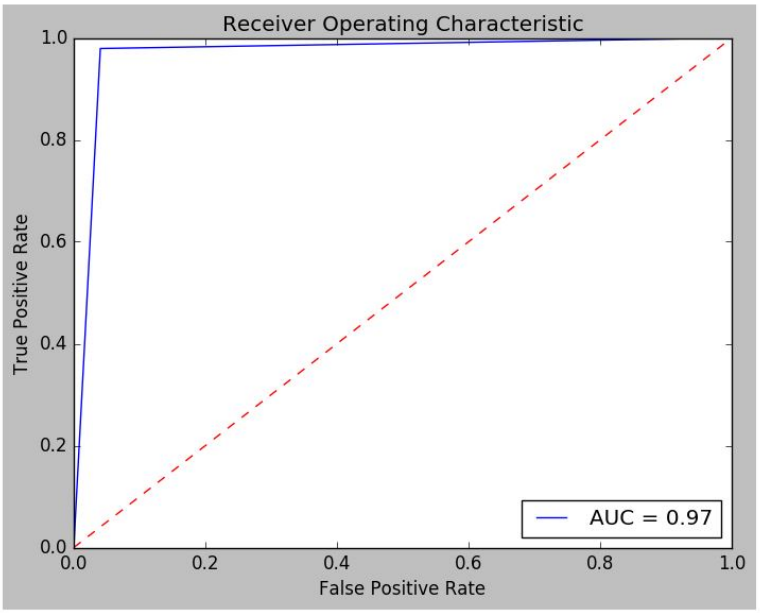

Figure 20: ROC and AUC of Logistic Regression: L2 , lambda = 1.0

```
accuracy: 0.969523809524
Confusion Matrix
[[1497   63]
 [  33 1557]]
```

```
==========Logistic Regression: L2 , lambda = 10.0==========
                      precision    recall  f1-score   support

  Computer technology       0.98      0.94      0.96      1560
Recreational activity       0.95      0.98      0.96      1590

          avg / total       0.96      0.96      0.96      3150

accuracy: 0.960317460317
Confusion Matrix
[[1470   90]
 [  35 1555]]
```

```
==========Logistic Regression: L2 , lambda = 100.0==========
                      precision    recall  f1-score   support

  Computer technology       0.99      0.89      0.94      1560
Recreational activity       0.91      0.99      0.95      1590

          avg / total       0.95      0.95      0.94      3150
```

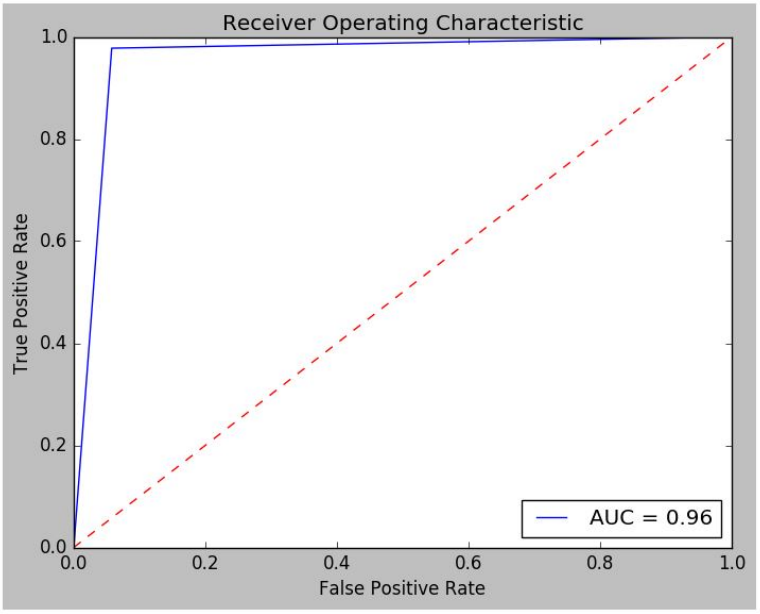

Figure 21: ROC and AUC of Logistic Regression: L2 , lambda = 10.0

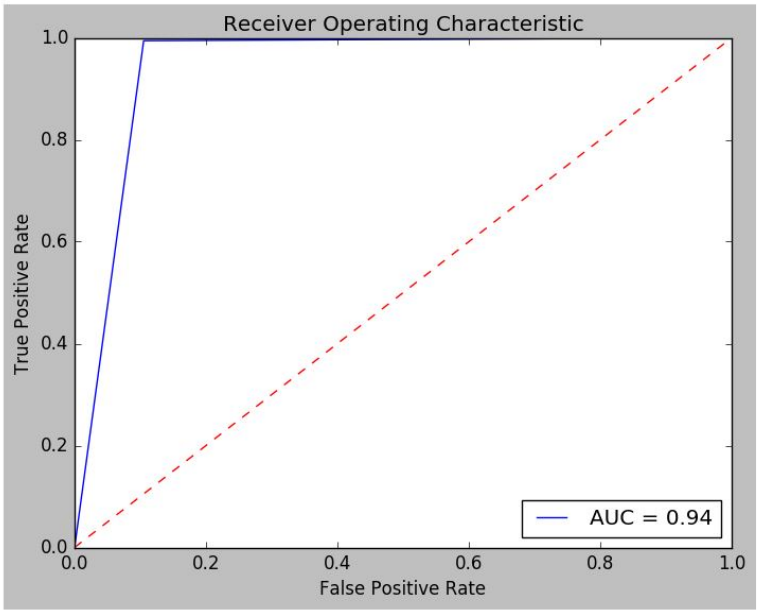

Figure 22: ROC and AUC of Logistic Regression: L2 , lambda = 100.0

```
accuracy: 0.945079365079
Confusion Matrix
[[1396  164]
 [   9 1581]]
```

5 Multiclass Classification:

5.1 Question j

Multiclass Classification can be done using the Naive Bayes Algorithm or the SVM algorithm. For this question we use the data from the following categories for multiclass classification.

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

Naive Bayes:

```
==========Multiclass Naive Bayes==========
                          precision    recall  f1-score   support

comp.sys.ibm.pc.hardware       0.70      0.82      0.76       392
   comp.sys.mac.hardware       0.92      0.63      0.74       385
           misc.forsale       0.74      0.84      0.79       390
  soc.religion.christian       0.96      0.97      0.97       398

             avg / total       0.83      0.82      0.81      1565

accuracy: 0.815974440895
Confusion Matrix
[[322  14  52   4]
 [ 88 241  51   5]
 [ 49   7 329   5]
 [  0   0  13 385]]
```

SVM :
One-vs-All, is implemented in OneVsRestClassifier. The strategy consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency (only n classes classifiers are needed), one advantage of this approach is its interpretability. Since each class is represented by one and only one classifier, it is possible to gain knowledge about the class by inspecting its corresponding classifier. This is the most commonly used strategy and is a fair default choice.

```
==========Multiclass SVM One Versus Rest==========
                          precision    recall  f1-score   support

comp.sys.ibm.pc.hardware       0.83      0.86      0.84       392
   comp.sys.mac.hardware       0.87      0.85      0.86       385
            misc.forsale       0.90      0.89      0.89       390
  soc.religion.christian       0.99      0.98      0.99       398

             avg / total       0.90      0.90      0.90      1565

accuracy: 0.896485623003
Confusion Matrix
[[338  35  18   1]
 [ 40 326  18   1]
 [ 28  12 348   2]
 [  3   0   4 391]]
```

OneVsOneClassifier constructs one classifier per pair of classes. At prediction time, the class which received the most votes is selected. In the event of a tie (among two classes with an equal number of votes), it selects the class with the highest aggregate classification confidence by summing over the pair-wise classification confidence levels computed by the underlying binary classifiers. Since it requires to fit N * (N - 1) / 2 classifiers, this method is usually slower than one-vs-the-rest, due to its O(N^2) complexity. However, this method may be advantageous for algorithms such as kernel algorithms which don't scale well with n samples. This is because each individual learning problem only involves a small subset of the data whereas, with one-vs-the-rest, the complete dataset is used N times.

```
==========Multiclass SVM One Versus One==========
                          precision    recall  f1-score   support

comp.sys.ibm.pc.hardware       0.83      0.86      0.84       392
   comp.sys.mac.hardware       0.87      0.85      0.86       385
            misc.forsale       0.90      0.89      0.89       390
  soc.religion.christian       0.99      0.98      0.99       398

             avg / total       0.90      0.90      0.90      1565

accuracy: 0.896485623003
Confusion Matrix
[[338  35  18   1]
 [ 40 326  18   1]
 [ 28  12 348   2]
 [  3   0   4 391]]
```