EE 219 Project 1

Yifan Shu, Chengshun Zhang, Xuan Yang

Introduction

Classification refers to the task of identifying a category, from a predefined set, to which a data point belongs, given a training data set with known category memberships. In this project, we researched different methods for classifying the textual data.

We firstly preprocessed the textual data by tokenizing each document into words, and then transformed it into a document-item matrix, where each row represents a document and each column represents the number of occurrence of a term in each document. Then we further used TF-IDF score to finally determine our data. After that, we performed dimensionality reduction by using Latent Semantic Indexing (LSI) and Non-Negative Matrix Factorization (NMF) methods.

With the data, we applied several different classification algorithms, starting with binary classification. In this project, we used Support Vector Machine (SVM), Naive Bayes, and Logistic Regression under different hyper-parameters. To evaluate how well our algorithm performs, we plotted the ROC curve, calculated the confusion matrix, accuracy, recall, precision and F-1 score.

At last, we performed SVM and Naive Bayes multiclass classification (with both One VS One and One VS the rest methods) and performed the same evaluation metrics for these classification methods.

Question 1

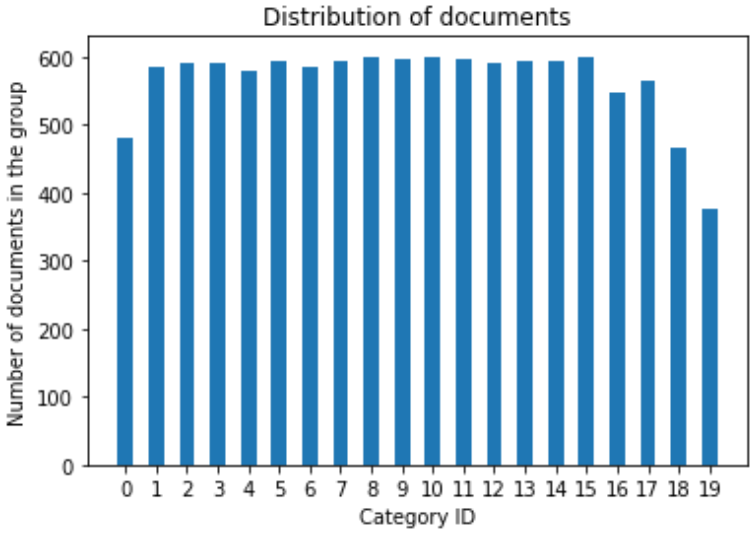

Figure 1. Histogram of the Number of Training Documents For Each Class

To make sure that the data is balanced, we plotted a histogram of the number of document in each topic, which is shown in Fig 1. The results shows that the sample is evenly distributed over the 20 classes. Therefore, there is no need to further balance the data.

Question 2

For this part, we first set the random seed to 42 to ensure the consistency. Then we performed the lemmatization with nltk.wordnet.WordNetLemmatizer and pos_tag. By defining our own tokenizing function, we can use CountVectorizer to get a document-term matrix which removes the 'english' stop words and numbers, and the words that show up less than 3 times. After that, we created a TF-IDF matrix using TfidfTransformer and fit the model with the training data. When min_df = 3, the number of terms is 16319.

The shape of train matrix is (4732, 16319).
The shape of test matrix is (3150, 11243).

Question 3

For this part, we used the TruncatedSVD and NMF modules provided in sklearn and set k = 50. Therefore, we can get the result of LSI and NMF.

$\|\|X - WH\|\|_F^2$	4106.962861236307
$\|\|X - U_k\Sigma_k V_k^T\|\|_F^2$	4141.549653993933

Table 1. The Loss for NMF and LSI

By calculating the $\|\|X - WH\|\|_F^2$ and $\|\|X - U_k\Sigma_k V_k^T\|\|_F^2$, which can be shown in Table 1, we find that the loss for NMF is larger. This is reasonable because according to eckart-young theorem, $U_k\Sigma_k V_k^T$ is the best k rank approximation for X. And WH sometimes cannot be written in the form $U_k\Sigma_k V_k^T$. Therefore, the loss for NMF is always larger or equal to that of LSI.

Question 4

(a) Compare the difference between hard margin SVM and soft margin SVM

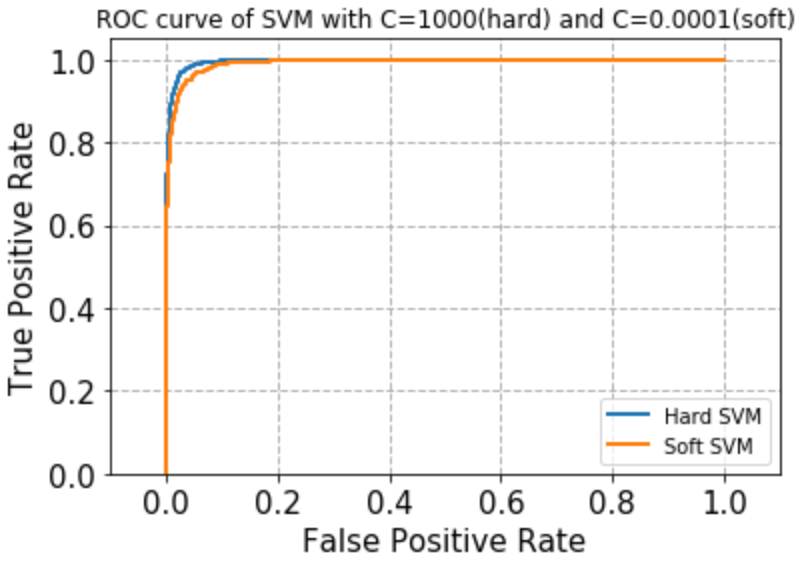

Figure 2. ROC Curve of SVM with C = 1000 (hard) and C = 0.0001 (soft)

	γ = 1000 (hard margin)	γ = 0.001 (soft margin)
Confusion Matrix	[[1504 56] [35 1555]]	[[0 1560] [0 1590]]
Accuracy	0.9711111111111111	0.5047619047619047
Recall	0.9779874213836478	1.0
Precision	0.9652389819987586	0.5047619047619047
F-1 Score	0.9715713839425181	0.6708860759493671

Table 2. The Performance of Hard Margin and Soft Margin Using Different Criteria

For this part, we plotted the ROC curve and calculated the confusion matrix, accuracy, recall, precision, and F-1 score. From ROC cure, it shows that hard margin SVM and soft margin SVM performs almost the same. But from confusion matrix, accuracy, recall, precision, and F-1 score, it is shown that hard margin SVM performs better.

This is because that, for soft margin, we set γ = 0.0001. For SVM problem, we are using numerical computation to find a solution. It is a linear optimization, thus is guaranteed to find out an optimal

solution. However, when γ = 0.0001, gradient around the optimal solution point will be small, thus, the numerical computation method may terminate without finding the final optimal solution, instead it will find a very close one. In this case, different from the expected SVM, ***ax+b*** > *0 implies that it is class* ***1***, and ***ax+b*** < *0* stands for class -1. However, if it is not an optimal solution, the decision boundary at 0 may not perform the best (e.g. the best result may be ***ax+b*** > *c implies that it is class* ***1***, and ***ax+b*** < *c*, where c != 0). Thus, for result of soft margin SVM shown in Table 2 can be actually corresponding to a point as Fig. 3.

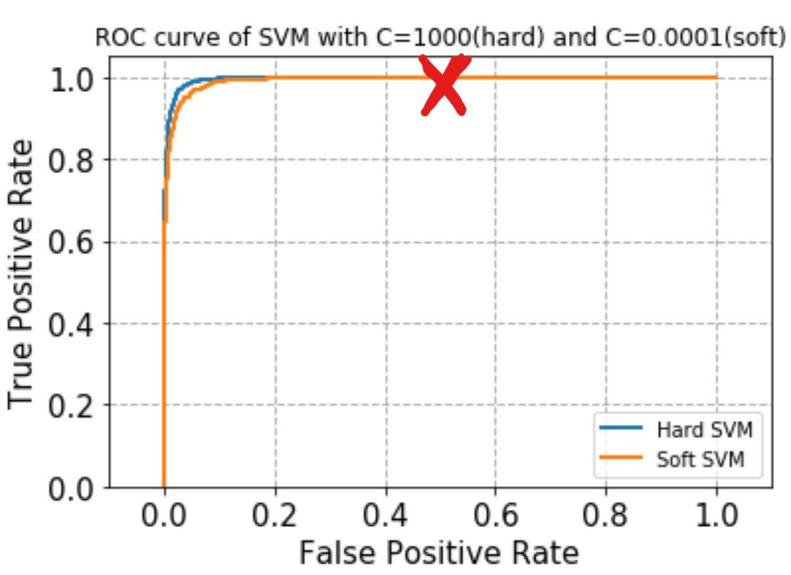

Figure 3. The Corresponding Point for the result of soft margin SVM shown in Table 2.

(b) Compare the difference between hard margin SVM and soft margin SVM

For this part, we are using mean accuracy to evaluate the performance for a SVM classifier. We set the parameter γ in the range of {0.001, 0.01, 0.1, 1, 10, 100, 1000}, and the mean accuracy for each γ can be shown in Table 3.

γ	F-1 Score
0.001	0.4946478873239437
0.01	0.4949295774647887
0.1	0.9642253521126761
1	0.969718309859155
10	0.971830985915493
100	0.9707042253521129
1000	0.9705633802816902

Table 3. The Mean Accuracy of SVM Classifier Using Different γ

From Table 3, it can be seen that $\gamma = 10$ performs the best for SVM. The corresponding ROC curve is shown in Fig. 4. Its confusion matrix, accuracy, recall, precision, and F-1 score is shown in Table 4.

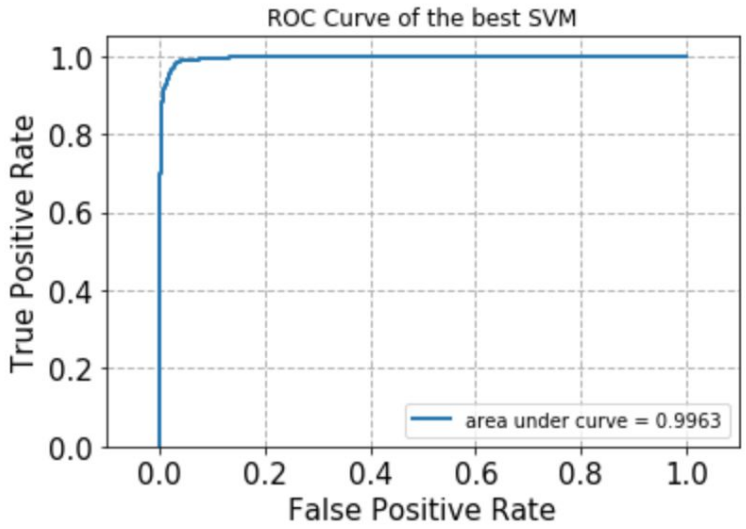

Figure 4. ROC Curve of SVM with C = 10 (soft)

	$\gamma = 10$ (soft margin)
Confusion Matrix	[[1512 48] [30 1560]]
Accuracy	0.9752380952380952
Recall	0.9811320754716981
Precision	0.9701492537313433
F-1 Score	0.975609756097561

Table 4. The confusion matrix, accuracy, recall, precision, and F-1 score of Soft Margin SVM Classifier Using Different γ

Question 5

(a) Train a Logistic Classifier without Regularization

We first tested the logistic classifier without regularization. The corresponding ROC curve is shown in Fig. 5. Its confusion matrix, accuracy, recall, precision, and F-1 score is shown in Table 5.

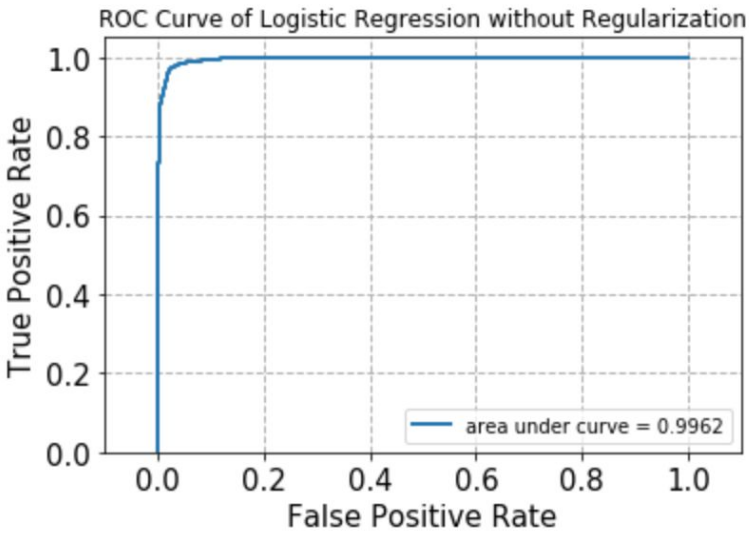

Figure 5. ROC Curve of Logistic Regression without Regularization

	Logistic Regression without Regularization
Confusion Matrix	[[1507 53] [32 1558]]
Accuracy	0.973015873015873
Recall	0.979874213836478
Precision	0.9671011793916822
F-1 Score	0.9734457981880662

Table 5. The Confusion Matrix, Accuracy, Recall, Precision, and F-1 Score of Logistic Regression without Regularization

(b) Train a Logistic Classifier with Regularization

For this part, we are using average test accuracy to optimize γ for a logistic regression classifier using 5-fold cross-validation, with both norm L1 and norm L2. Besides, we tried to find out the best γ (γ is in the range of {0.001, 0.01, 0.1, 1, 10, 100, 1000}) for both of the logistic regression classifiers. The result is shown in Table 6.

	Mean Accuracy of Logistic Regression with L1 Norm	Mean Accuracy Logistic Regression with L2 Norm
0.001	0.4977464788732394	0.6911267605633803
0.01	0.4977464788732394	0.9083098591549295
0.1	0.9567605633802817	0.9636619718309859
1	0.9656338028169014	0.9688732394366196
10	0.9740845070422536	0.9721126760563379
100	0.9729577464788732	0.973661971830986
1000	0.9728169014084507	0.9729577464788732

Table 6.. The Mean Accuracy of Logistic Regression Classifiers with L1 norm and L2 norm

Therefore, from Table 6, we can tell that the best γ for logistic regression classifier with L1 norm is 10 and the best γ for logistic regression classifier with L2 norm is 100. Thus, in this part we tried to compare the following three classifiers, logistic regression classifier without regularization, logistic regression classifier with L1 norm (γ = 10), and logistic regression classifier with L2 norm (γ = 100). We used the test data and used accuracy, precision, recall, and F-1 score to evaluate the performance. The result is shown in Table 7.

	Logistic Regression Classifier without Regularization	Logistic Regression Classifier with L1 Norm (γ = 10)	Logistic Regression Classifier with L2 Norm (γ = 100)
Accuracy	0.973015873015873	0.9720634920634921	0.9733333333333334
Recall	0.979874213836478	0.979874213836478	0.9811320754716981
Precision	0.9671011793916822	0.9653035935563816	0.966542750929368
F-1 Score	0.9734457981880662	0.9725343320848939	0.9737827715355806

Table 7. The Performance of Logistic Regression Classifier without Regularization, Logistic Regression Classifier with L1 Norm (γ = 10), and Logistic Regression Classifier with L2 Norm (γ = 100)

As we can see from the result, the larger the regularization parameter, the less the test error. Besides, we check the learnt coefficients, the learnt coefficient can be shown in Table 8.

	Learnt Coefficients
Logistic Regression Classifier without Regularization	[[-4.82227468 124.04432871 -25.16035881 91.53510578 14.6348396 -17.44440009 -3.434508 1.51007109 24.0498891 18.3969862 33.45209612 -4.58073603 -12.29631953 11.73037924 -16.70782175 -2.57166256 -10.72619183 13.46550213 16.53555859 3.34556521 -0.89393895 -1.28185099 -4.37428383 5.91357888 10.01942029 -3.52488137 16.88711953 -6.90756099 -12.79918896 21.17840376 1.47271195 -10.74987683 11.59541362 -4.45628672 -7.27049459 1.64818731 10.14868016 -11.84525038 3.75176382 8.47487332 6.4412368 15.76628075 -11.60271396 3.92011692 2.62737915 -5.56168779 -11.21385135 4.0024145 7.98094118 -5.18590256]]
Logistic Regression Classifier with L1 Norm (γ = 10)	[[-2.04715848 106.2594828 -18.1330126 73.90277308 11.4063873 -11.25435226 -1.64745972 0. 18.07938578 15.31649281 27.28744345 -4.02875985 -9.33645304 6.87199647 -13.4188939 0. -7.07889185 7.90094613 12.63623293 2.41058466 0. 0. -3.84509294 2.8681963 6.34291341 -1.18460636 13.85088223 -4.46930875 -9.29987336 17.03016534 0. -5.69055324 7.45445359 -1.2289182 -3.15664312 0.7260542 6.80966594 -8.70932238 1.72307869 7.2052753 5.19214443 11.93539065 -7.87358252 0.64107627 0. -4.30909045 -8.05268962 1.63897863 4.4200187 -1.92986537]]
Logistic Regression Classifier with L2 Norm (γ = 100)	[[-1.80574391 84.4162102 -15.19463256 58.80136108 8.56720101 -8.58725551 -2.35819941 -0.1795704 14.97082934 13.60928514 22.74658521 -4.84533522 -7.96722821 6.47179954 -9.88313175 -1.93067748 -6.44336423 9.35303303 9.70361268 2.97531115 -0.56287474 -1.20049871 -3.94668122 2.02709011 8.4714677 -2.94014203 9.68399684 -4.27898045 -7.33054535 13.92009648 0.61335404 -5.44999424 5.44989476 -1.14252567 -3.75014237 1.25088043 4.97317518 -6.0489856 1.00255284 6.14734352 4.30216972 8.23015096 -7.34139904 0.71762537 0.75663648 -4.15753854 -6.60890191 1.9478189 4.29269555 -1.71441291]]

Table 8. The Learnt Coefficients of Logistic Regression Classifier without Regularization, Logistic Regression Classifier with L1 Norm (γ = 10), and Logistic Regression Classifier with L2 Norm (γ = 100)

From Table 8, it is shown that the learnt coefficients of logistic regression classifier without regularization may contains very big number. The learnt coefficients of logistic regression classifier with L1 norm (γ = 10) may contains a lot of zeros and do not contains very large number. The learnt coefficients of logistic regression classifier with L2 norm (γ = 100) is similar to that of logistic regression classifier without regularization, but does not contains very large number.

This findings can be very useful when determining how we should choose hyper-parameter. For both L1 norm and L2 norm, the larger the regularization term, the less the learnt coefficients ten do be. Besides, there is significantly difference between L1 norm and L2 norm. If the ground truth coefficients is sparse

(contain a lot of 0), in this case, L1 norm may perform better. Because the logistic regression classifier with L1 norm tend to learn sparse coefficients.

(c) Differences Between Logistic Regression and SVM

Though logistic regression and linear SVM are trying to classify data points using a linear decision boundary. But SVM is trying to determine the boundary with maximal margin, by solving its dual problem. Logistic regression is different, it tries to maximize the likelihood function of all the data point, by using gradient descent methods.

The performance differ because of the difference of finding the decision boundary. SVM usually performs better when because it is a large margin classifier. However, SVM may perform worse when the dataset is small, thus data points near the decision boundary may not be true representation of the actual decision boundary, thus may form false maximum margin classifier boundary.

Question 6

In this part, we applied Naive Bayes classifier and trained a GaussianNB classifier. The corresponding ROC curve is shown in Fig. 6. And its confusion matrix, accuracy, recall, precision and F-1 score are also shown in the following Table 6.

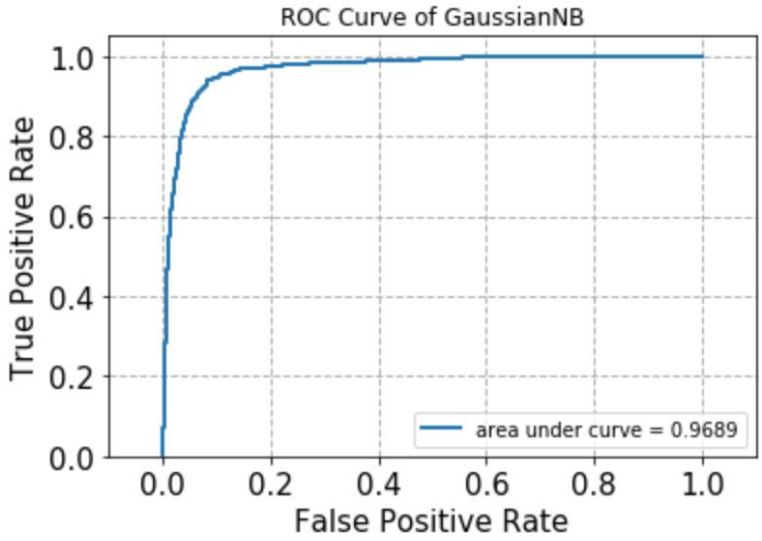

Figure 6. ROC curve of GaussianNB

	GaussianNB classifier
Confusion Matrix	[[1316 244] [50 1540]]

Accuracy	0.9066666666666666
Recall	0.9685534591194969
Precision	0.8632286995515696
F-1 Score	0.9128630705394191

Table 6. The Confusion Matrix, Accuracy, Recall, Precision and F-1 Score of GaussianNB classifier

Question 7

We first constructed a pipeline that performs feature extraction with min_df=3 and remove English stop words and pure numbers; dimensionality reduction with LSI; linear SVM classifier that has parameter γ=100. Fit the pipeline to the training dataset we got the accuracy of this pipeline to be 97.36%.

We did grid search with the options shown in Table 7.

Procedure	Options
Loading Data	Keep “headers” and “footers”
	Remove “headers” and “footers”
Feature Extraction	min_df=3, without lemmatization
	min_df=5, without lemmatization
	min_df=3, with lemmatization
	min_df=5, with lemmatization
Dimensionality Reduction	LSI
	NMF
Classifier	SVM with γ=10
	Logistic Regression: L1 Regularization with C=10
	Logistic Regression: L2 Regularization with C=100
	GaussianNB

Table 7. Options to compare

Note that we did not remove number without lemmatization. In practice, we did one grid search without removing “headers” and “footers” and got 32 different combinations. Then we did another grid search with “headers” and “footers” removed and got another 32 combinations. So the total different combination is 64. The detailed result shown in the format of pandas table is in the notebook. From the result, we found that the best combination is:

- Keep “headers” and “footers”
- min_df=5 with lemmatization
- LSI
- Logistic Regression with L1 regularization

Question 8

In this part, we applied Naive Bayes, SVM on One VS One method and SVM on One VS the Rest method to learn documents belonging to multiple classes, which is different from binary classification.

For Naive Bayes classification, we still used GaussianNB classifier. For multiclass SVM classification on One VS the Rest method, since it’s still linear classification, we used the best SVM classifier again (kernel = “linear”, C = 100) and the default multi-class classification method is already One VS the Rest. For multiclass SVM classification on One VS One method, it’s supported by SVC classifier. So we used SVC classifier with “kernel” parameter set to “linear” and “decision_function_shape” parameter set to “ovo”. In order to compare with SVM on One VS the Rest method, we also set its C parameter to 100.

To calculate recall, precision and F-1 score for multi-classification, one more parameter needs to be set. Since we’ve already known from Question 1 that the data set of the categories we mainly worked on is already balanced, we do not need to consider about label imbalance. Therefore, we set the parameter “average” for these evaluation values to “macro” to calculate metrics for each label, and find their unweighted mean. The results are as the following Table 8:

	GaussianNB	SVM (ovr)	SVM (ovo)
Confusion Matrix	[[232 38 120 2] [107 159 118 1] [48 45 295 2] [0 1 15 382]]	[[305 61 26 0] [44 317 24 0] [24 20 343 3] [3 0 2 393]]	[[305 61 26 0] [44 317 24 0] [24 20 343 3] [3 0 2 393]]
Accuracy	0.6824281150159744	0.8677316293929712	0.8677316293929712
Recall	0.6802582497665053	0.8670905533208118	0.8670905533208118
Precision	0.6948013657577683	0.8671078244075621	0.8671078244075621
F-1 Score	0.6760627830593133	0.8669467187208237	0.8669467187208237

Table 8. Evaluation scores for multi-class classifiers

From the results above, we could see that the performance of SVM (ovo) and SVM (ovr) is similar but SVM (ovr) is a little better. But they are all greatly better than GaussianNB classifier.