LDA and Truncated SVD in Document Multi-classification: Application and Comparison

Zhixuan Qin, Math 253 Spring 2020

Abstract: Document classification assigns text documents to predefined categories, which helps the organization and retrieval of massive text data. Dimensionality reduction of high-dimensional text dataset is an important preprocessing step for classification. In this project, two popular methods, Truncated Singular Value Decomposition (SVD) and Linear Discriminant Analysis (LDA), were used to reduce the dimensionality of a big text dataset with five categories. Our results showed that LDA-based reduction helps to separate samples from different categories when data was projected to a low-dimensional subspace and contributed to better classification performances compared with SVD-based reduction. Overall speaking, the selection of reduction method is a trial and error and should also consider the data storage size (related to the final reduced dimension) and the cost of running time.

Introduction

With the increased popularity of the Internet, the amount of text data has grown exponentially in the past few decades. As a result, the storage and organization of the text data has become very challenging with the information overload. A good way to organize such massive text data is to categorize the text data by assigning predefined categories to each given text document based on the content. Once text is categorized, it will also help process such as information retrieval, text dissemination and filtering when data is needed for future use. However, text data typically contains high-dimensional features with hundreds or thousands of feature space. The major challenge to categorize the text data is to efficiently reduce the dimensionality of the data features to improve the effectiveness of classification and reduce storage requirement.

Two popular methods for dimensionality reduction are Truncated Singular Value Decomposition (SVD) and Linear Discriminant Analysis (LDA). The former is an unsupervised (i.e., does not recognize label information) method to approximate a sample data with selected principal components[1], while the latter is a supervised (i.e., recognize class information) method that preserves discriminatory information between different classes of a dataset[2]. Truncated SVD is very similar to Principal Component Analysis (PCA), which reduces the dimensionality of the data features by finding the maximum variance directions. However, the truncated SVD applies the matrix factorization technique on the data matrix, whereas the PCA applies the factorization on the covariance matrix. As a result, one advantage of truncated SVD over PCA for text data dimensionality reduction is that the truncated SVD can be applied to sparse matrix, an efficient and popular format to store text data. Application of PCA on sparse matrix is impossible because covariance matrix must be computed, which requires the operation on the entire matrix. Linear discriminant analysis is a popular dimension reduction method for pattern recognition[2]. It has been widely used for dimensionality reduction before classification. It helps to find a linear combination of features that best separates samples from two or more classes. The objective of this project is to apply truncated SVD and LDA to reduce the dimensionality of a big dataset and evaluate and compare their effectiveness/efficiency in text classification.

Review of Mathematical Theory

<u>Truncated SVD</u>

For any matrix A of dimension n×m, there exist two orthogonal matrices U of dimension n×n, V of dimension m×m, and a nonnegative diagonal matrix Σ of dimension n×m, such that A can be shown as:

$$A = U\Sigma V^T$$

The matrix Σ stores the singular values (σ_i) of A at its diagonal in a decreasing order. The columns of U (u_i) and V (v_i) are the left and right singular vectors of A, respectively. As a result, the matrix A with rank r (r<= min(n,m)) can be considered as a weighted sum of a series of rank-one matrices:

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

We can approximate A by using selected k (normally 1<=k < =r) largest singular values and corresponding left and right singular vectors, which directly reduces the dimension of matrix A to k principal components:

$$A \approx \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

LDA

Linear discriminant analysis is a popular linear supervised (i.e., with label information) dimension reduction method. It aims to discriminate data from different classes by maximizing the between-class scatter and minimizing the within-class scatter. The objective function of LDA is:

$$\max_{v:||v||=1} \frac{v^T S_b v}{v^T S_w v}$$

S_w and S_b is the within-class and between-class scatter matrix, respectively and is computed as:

$$S_w = \sum_{i=1}^{c} \sum_{j=1}^{n_i} (x_j - \mu_i)(x_j - \mu_i)^T$$

$$S_b = \sum_{i=1}^{c} n_i (\mu_i - \mu)(\mu_i - \mu)^T$$

Where c is the number of classes, n_i is the number of samples within the i-th class, $\mu_i = (\frac{1}{n_i}) \sum_{i=1}^{n_i} x_j$ is the sample mean of the ith class, and $\mu = (\frac{1}{n}) \sum_{i=1}^{n} x_i$ is the overall sample mean.

The objective function of LDA can be solved by finding the eigenvectors associated with the largest eigenvalues of the following generalized eigenvalue problem:

$$S_b v = \lambda S_w v$$

If S_w is nonsingular, we can solve the above problem by computing the leading eigenvectors of $s_w^{-1}S_b$. If S_w is singular, other reduction methods, such as truncated SVD, can be applied first to reduce the dimensionality of the labeled data before the application of LDA. In either case, there are at most c-1 eigenvectors by LDA method since the rank of S_b is bounded by c-1.

A Brief Introduction to Support Vector Machine

The Support Vector Machine (SVM) has been recognized as one of the most successful classification algorithms and has been widely used for text classification[2]. It is a supervised learning algorithm based on the principal of maximal margin bound[2], i.e. the maximum distance between data points of two classes.

Assume we have n points that belong to two different classes:

$$\{(x_i, y_i)\}_{i=1}^{n}, y_i \in \{-1, +1\}$$

where x_i is an l-dimension vector and y_i is the label information of classes that the vector belongs to. The SVM separates the two classes by a hyperplane with a decision function[2]:

$$w^T x + b = 0$$

where x is an input vector, w is an adaptive weight vector, and b is a constant (bias). To find the maximum-margin hyperplane, we need to find[2]:

$$\min_{w,b} \frac{1}{2} w^T w$$

subject to the constraint that:

$$y_i(w^T x_i + b) \geq 1$$

using Lagrange multipliers approach, the above minimization problem can be solved as[2]:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \bullet x_j)$$

with the constraints that $\alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i y_i = 0$.

This x_i is called support vector when α_i does not equal to 0 and contributes to the decision function. It can also be shown that the decision function can be derived as[2]:

$$f(x) = sgn\left(\sum_{i=1}^{n_{sv}} \alpha_i y_i (x_i \bullet x) + b\right)$$

where n_{sv} is the number of support vectors.

Normally, a linear hyperplane can be used to separate samples from different classes. In the case that the data has nonlinear separation relationship, we can use the kernel trick to perform a nonlinear mapping so that we can transform the data into a new feature space[2]:

$$x_1 \bullet x_2 \rightarrow \varphi(x_1)^T \varphi(x_2) = K(x_1, x_2)$$

where K(•) is a positive definite kernel function.

After the implementation of kernel function, the SVM decision function becomes[2]:

$$f(x) = sgn\left(\sum_{i=1}^{n_{sv}} \alpha_i y_i K(x_i \bullet x) + b\right)$$

Materials and Methods

Testing Dataset

The HuffPost news dataset (https://www.kaggle.com/rmisra/news-category-dataset) contains approximately 200,000 news from the past 5 years. Each news record contains information including category, headline, authors, link to the post, publication date, and a short description of the article. The link to the post was used to access the full text of each article. For this project, five categories were selected (a total of 26194 news documents) and the number of documents in each category was displayed in Figure 1.

Implementation Procedure

This project was computed using Python 3 on Jupyter Notebook. Raw documents were first converted to lowercase characters and cleaned through Natural Language Toolkit (NLTK; https://www.nltk.org/) to remove all the special characters (e.g., punctuation, comma, etc.), stop words (e.g., "the", "a", etc.) and trivial words. Documents were further trimmed to their stem through NLTK to reduce the vocabulary. Additionally, words that appeared in more than 90% (document frequency>23,575) or less than 0.1% (document frequency<262) of the total documents were removed. The cleaned documents were then converted to feature vectors by the term frequency-inverse document frequency (TFIDF) method[3] and the corresponding labels were included in a separate vector file. The entire dataset was split into training and testing datasets using Scikit-learn model selection library[4]. Specifically, for each category, 80% of the samples were randomly selected as the training data and the remaining 20% were selected as the testing data.

Different combinations of the LDA and truncated SVD algorithms were applied to the training and testing data to reduce their dimensionality in feature spaces. Scikit-learn decomposition[5] and discriminant analysis[6] libraries were used to implement truncated SVD and LDA, respectively. Specifically, we tested the dimensionality reduction via truncated SVD alone (reduced to 6 or 3 dimensions), 90, 70, 60, or 10% truncated SVD (the percentages indicate the explained variance after the dimensionality reduction by truncated SVD based on the training dataset) + 3 or 4 components LDA.

The support-vector machines (SVM) classifier was trained based on the reduced training dataset using Scikit-learn SVM library[7]. The SVM classifier with linear kernel type was implemented with one-vs-rest classification type. The trained classifier was later used to predict labels to the reduced testing dataset.

The running time of different dimensionality reduction methods were compared using the "%timeit" function. The impacts of different dimensionality reduction methods to classification performance were visualized and compared by receiver operating characteristic (ROC) curve[8].

The ROC curve typically plots the true positive rate (i.e., correct prediction) on the y axis and the false positive rate (i.e., wrong prediction) on the x axis. A good prediction method would yield points in the upper left corner of the ROC curve space and a large area under the curve (AUC) score.

Results and Discussion

Reduction of dimensionality

After being converted to feature vectors, all the training and testing documents had a final vector dimension of 10,800. Since the data matrices were highly sparse, truncated SVD was used before the application of LDA to solve the singularity issue of within-class scatter matrix. Overall speaking, dimensionality reduction of training data matrix by a combination of truncated SVD and LDA method resulted in good separations among categories in the 3-dimensional space (Figure 2 a-c). The 90 or 60% truncated SVD + 3 components LDA seems to result in a slightly better separation among classes compared with the 10% truncated SVD + 3 components LDA. But when only truncated SVD was used to project the data into 3-dimensional space, samples among different categories were mingled together (Figure 2 d). This is majorly because the unsupervised nature of truncated SVD and the fact that we lost a lot of information (explained total variance is only 2%) when only 3 principle components were selected to represent the data.

Runtime efficiency of dimensionality reduction

The running time of dimensionality reduction was evaluated for 3 components truncated SVD, 10, 60, 70, or 90% truncated SVD + 3 components LDA (Figure 3). Dimensionality reduction by 3 components truncated SVD was the fastest since it is a one-step reduction approach while methods involving both truncated SVD and LDA took longer running time. We also observed that when more components were kept for truncated SVD to explain higher total variance, the running time of dimensionality reduction increased dramatically (Figure 3). This will make a significant difference in terms of runtime efficiency when text documents with high-dimensional feature space are needed for dimensionality reduction.

Classification performance

We observed better classification performance based on 6 components truncated SVD (Figure 4 a) compared with the classification performance based on 3 components truncated SVD (Figure 4 b), since more components preserved more information for accurate classification. The LDA-based dimensionality reduction (4 components) yielded near perfect classification results with AUC score close to 1(Figure 4 c-f), which is consistent with the fact that LDA-based method resulted in good separations among categories in the 3-dimensional space (Figure 2 a-c). It is worth to mention that 10% truncated SVD + 4 components LDA contributed to classification performance as good as 90% truncated SVD + 4 components LDA. However, the running time with 90% truncated SVD can be much higher than 10% truncated SVD (Figure 3), which should be considered when deciding the methods for dimensionality reduction. Overall speaking, with the tested parameter settings, LDA-based method outperformed SVD-based method for classification accuracy in our experiment. However, it is likely that SVD-based method with more dimensions (i.e., more components) can contribute to more accurate classification results.

We noted that the class dimensions were inconsistent during the dimensionality reduction (5 classes) and classification (2 classes) processes in our implementation. In other words, the

features generated by 3 components/4 components LDA was optimized to maximize the separation among 5 classes, which might not be the ideal features for one-vs-rest binary classification. Thus, we revisited the 10% truncated SVD + 4 components LDA approach and focused on categories of business and healthy living. In this new trial, instead of using dimension-reduced training data generated by 4 components LDA with 5-class labels to train all one-vs-rest binary classifiers, we applied 1 component LDA to the training data for each one-vs-rest classifier. For example, for the business-vs-rest classifier, we considered other categories other than business as 1 category (i.e., rest), and applied 1 component LDA following 10% truncated SVD to reduce the dimensionality of the training dataset. Using this approach, we clearly saw a good separation between the business category versus the rest groups in the 1-dimensional space (Figure 5 a). The classification result (Figure 5 b) was the same as the result from the previous implementation (Figure 4 f; AUC=0.98). We repeated the same process for the healthy living-vs-rest classifier, clear separation in the projected space (Figure 5 c) and the same classification performance as before (Figure 5 d; AUC=0.97) were also observed. As a result, these two approaches were equally effective with our data.

Conclusions

Dimensionality reduction is a common preprocessing step for classification, which helps to learn a better classifier and increase classification efficiency by learning a classifier on low-dimensional inputs. Depending on the characteristics of the input data (e.g., linear vs nonlinear separation relationship) and the settings of the classifiers, various dimension reduction methods with different parameter settings should be compared and evaluated to select the most appropriate one for classification preprocessing. The data storage size (related to the final reduced dimension) and the cost of running time should also be considered when deciding the methods for dimensionality reduction.

Lessons Learned

I presented contradictory results during the class presentation with LDA-based method contributed to poor classification performance (even worse than the results based on 3 components truncated SVD). It turns out that when I reduce the dimension of training and testing dataset with SVD/LDA, I used fit_transform function in Python, which finds the optimal feature spaces (either to maximize variance or separation depending on the selected method) and projects the data onto the optimal spaces. As a result, my training and testing dataset were projected onto different spaces (whatever optimal for training and testing dataset respectively). The correct way to do this (as I presented in this report) is to use the fit_transform function only on the training dataset to find the optimal spaces based on the training data and then project the testing data onto the same spaces using the transform function. Since LDA is a supervised algorithm that uses more information from the data, classification performance based on LDA was affected the most.

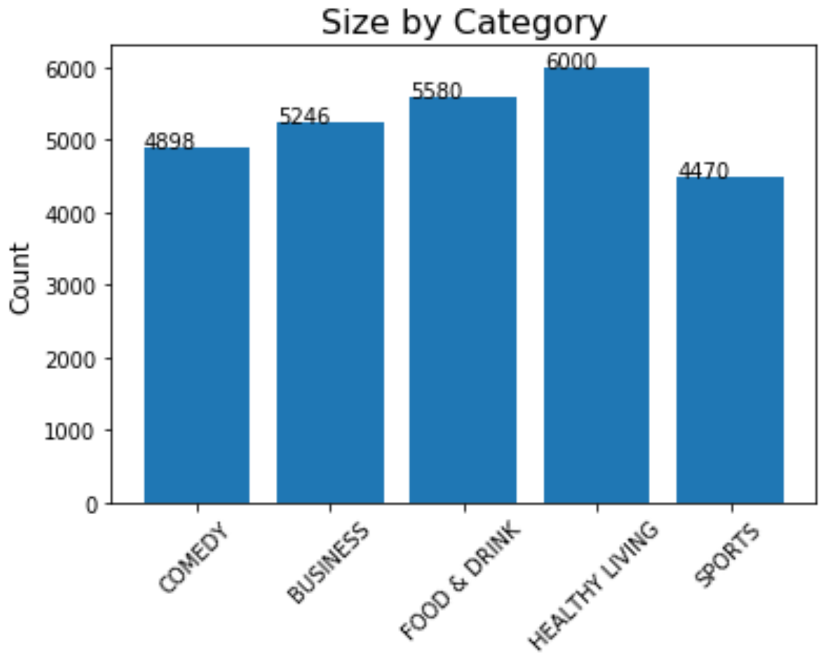

Figure 1. Number of documents by category.

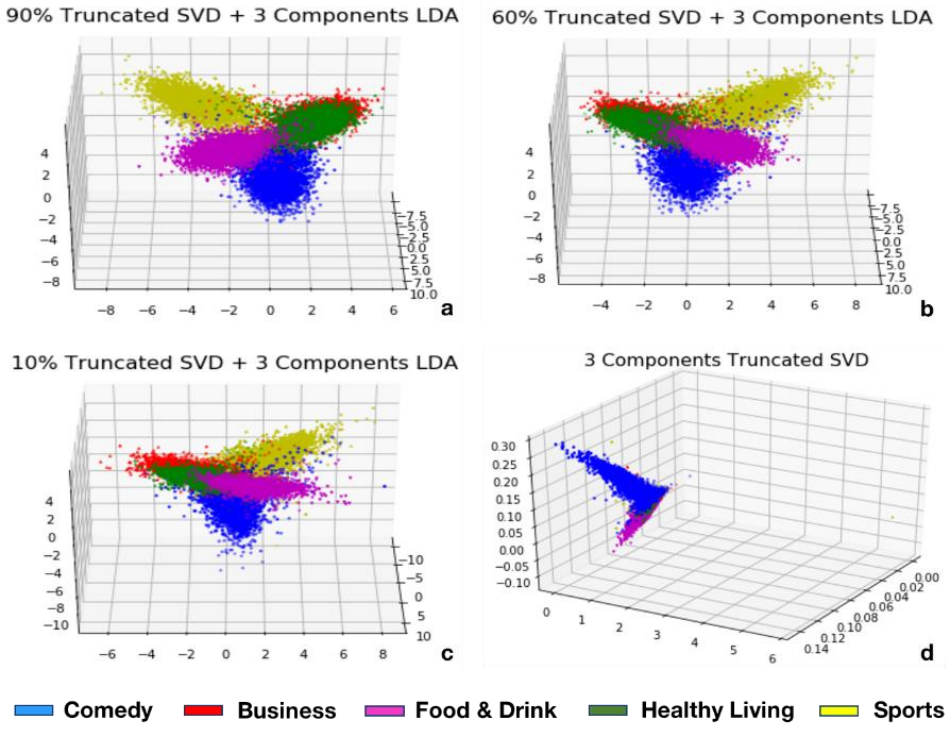

Figure 2. Projection of training dataset into 3-dimensional space after dimensionality reduction by different methods involving SVD and LDA.

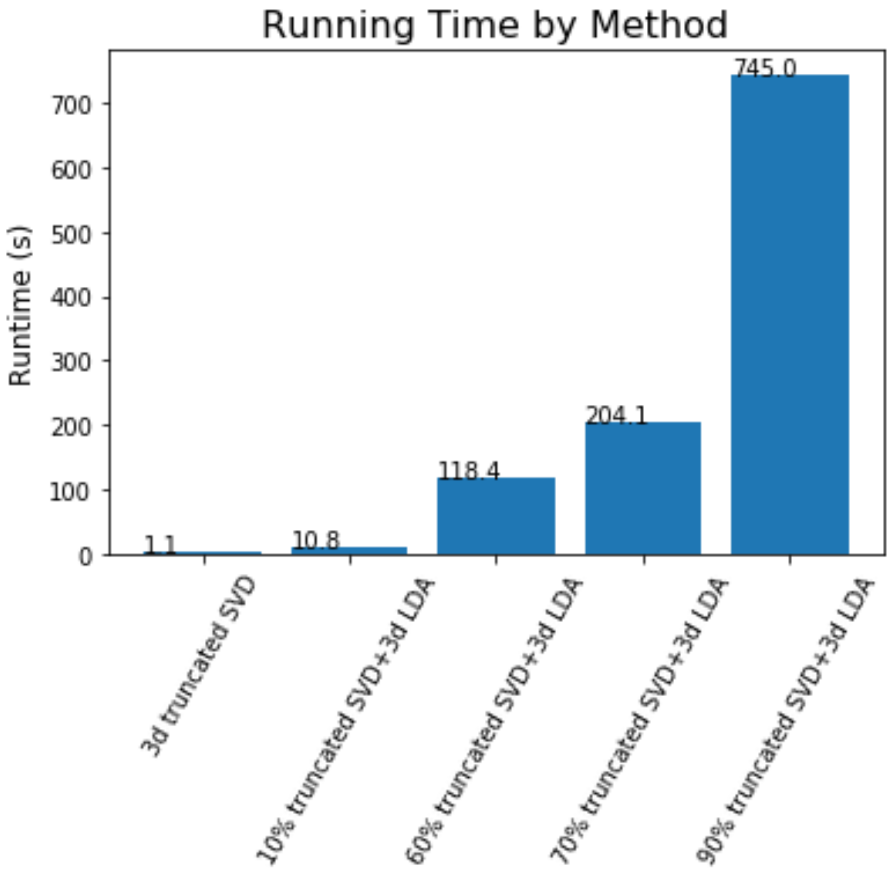

Figure 3. The running time in seconds of dimensionality reduction by different methods involving truncated SVD and LDA.

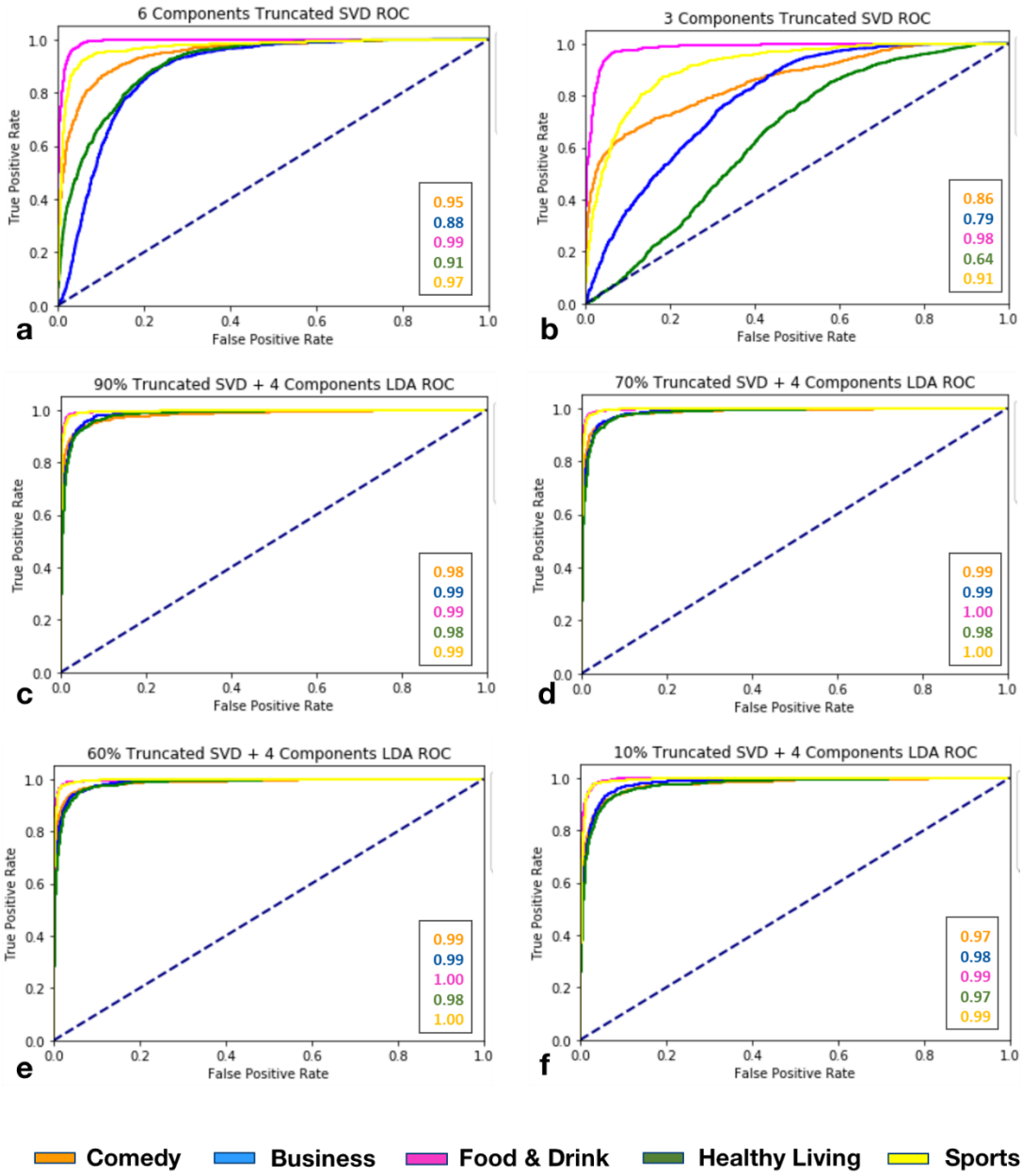

Figure 4. Visualization of classification performance by receiver operating characteristic (ROC) curve when different methods involving truncated SVD and LDA were used for dimensionality reduction. The numbers framed in each graph represent the area under the curve (AUC) score of each color-coded category.

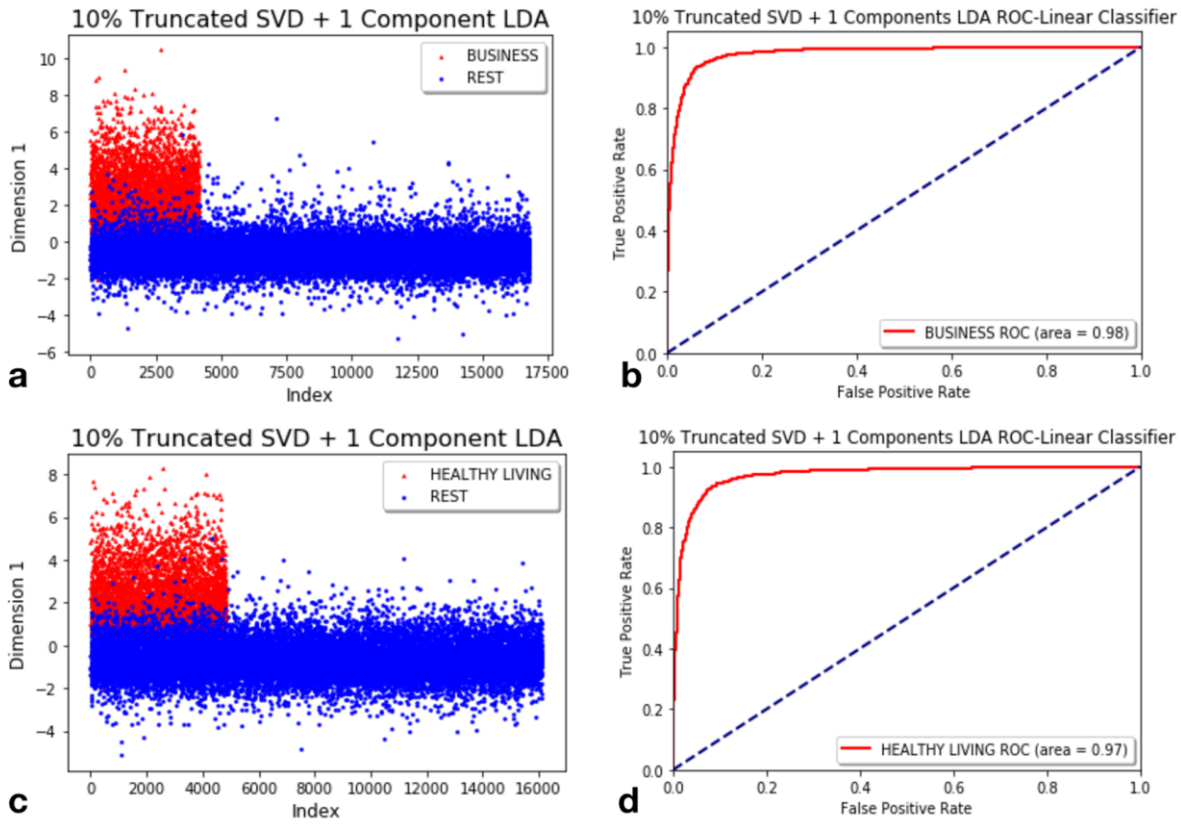

Figure 5. Projection of category of business and healthy living based on 10% truncated SVD + 1 component LDA, and the corresponding classification results when data was grouped as one-vs-rest pairs.

Reference

1. Zhang, Z., and Zha, H. (2004). Linear low-rank approximation and nonlinear dimensionality reduction. *Science in China Series A: Mathematics*. 47, 908-920.
2. Wang, Z. and Qian, X. (2008). Text categorization based on LDA and SVM, *Computer Science and Software Engineering, 2008 International Conference on*, Vol. 1, IEEE, pp. 674–677.
3. Rajaraman, A. and Ullman, J.D. (2011). Data Mining. *Mining of Massive Datasets*. pp. 1-17. doi:10.1017/CBO9781139058452.002.
4. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
5. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html
6. https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html
7. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
8. https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html
9. Tharwat, A., Gaber, T., Ibrahim, A., and A.E. Hassanien. (2017). Linear Discriminant analysis: a detailed tutorial. *Ai Communications* 30(2): 169-190
10. Mikolov, T., K. Chen, G. Corrado, and J. Dean. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv*:1301.3781, 2013.
11. Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, vol.34, pp.1-47, March 2002.