Text Classification Using Latent Semantic Analysis with Sparse Principal Component Analysis

by

Yifei Gao

A thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Science

Courant Institute of Mathematical Sciences

New York University

September, 2022

Jonathan Goodman

Abstract

This paper mainly focuses on the application of latent semantic analysis to text messages from Chinese social media platform Weibo using sparse principal component analysis (SPCA), which provides a more easily interpretable sparse matrix with faster calculations, in comparison to ordinary principal component analysis (PCA). The causes of the disparities in the LSA with PCA and the LSA with SPCA are investigated, and discussions on the limitation of LSA with SPCA are presented.

Contents

Abstract iii

List of Figures vi

1 Introduction 1

1.1 Background 1

1.2 Paper structure 3

2 Data Cleaning 4

2.1 Data description 4

2.2 Word separation 4

2.3 Feature extraction 6

2.4 Removing insignificant words 7

2.5 Removing bot-generated feeds 7

2.5.1 Measures of similarity 9

3 Latent Semantic Analysis with PCA and SPCA 11

3.1 Vectorization of raw data 11

3.1.1 Applying TF-IDF vector technique 11

3.2 Usage of principal component analysis 12

3.3 Usage of sparse principal component analysis 14

3.4 Classification using K-means clustering method 16

4 Evaluation of the Empirical Results 18

4.1 Empirical results of SPCA and PCA from K-means 18

4.2 Further experiment on SPCA and PCA 20

4.2.1 Evaluation metrics 20

4.2.2 Supervised learning classifier methods 21

4.2.3 Empirical results with supervised learning classifier methods 21

5 Conclusion 24

Bibliography 26

List of Figures

1 First 5 Rows from the Data Set. 5

2 First 5 Rows from the Data Set with the "separated_content" column. . . . 6

3 Table with all words and their frequencies. 8

4 Example of Suspicious bot-generated feeds 10

5 IVM with TF-IDF Filtration 12

6 Alogrithm for TF-IDF Technique 13

1 Introduction

1.1 Background

There is a huge amount of textual messages on social media platforms today. Great potential value is hidden in such textual messages: instant feedback from customers on social media can help improve products; real-time discussions can reveal general attitudes towards trendy events; malicious propaganda from terrorist groups can possibly be exposed and deleted in time [8]. To extract these certain messages, effective organization and management are in high demand. Automatic text classification methods are being applied to categorize this amount of data into thematic categories [11]. In this study, I focus on exploring the performance of text classification using textual data from a social media platform called Weibo (微博). The further usage of the cluster result will leave readers to explore.

Text classification relies on either supervised learning or unsupervised learning approaches. For supervised learning approaches, text classification categorizes the textual objects through manual annotation with predefined labels [5]. The goal for supervised text classification is to predict labels for new textual data. For unsupervised learning approaches, text classification assigns labels based on the observation of data without a predefined set of labels. The goal of unsupervised learning is to get insights from large volumes of textual data and to determines what is different or interesting from the data set.

The high dimensionality of textual data is a problem for text classification. Social me-

dia users on Weibo produce millions of unstructured texts per day [12]. To analyze such textual data, text classification methods require working with high dimensional matrix of word embedding from the textual data. Word embedding is a natural language processing technique that converts texts and words into column vectors and row vectors, which I will explain in detail in Section 3.1. The high dimensional matrix, made up of column vectors and row vectors, is often sparse and include many zeros. The high dimensionality causes clustering methods to become time-consuming and difficult for training and implementing. Therefore, processing the high dimensionality of social media messages has become a growing topic in recent years. Latent semantic analysis (LSA) is one of the auxiliary methods for text classification to deal with high dimensional data. LSA was introduced to improve the information retrieval, reducing the dimensions of such sparse matrices [4].

Latent semantic analysis consists of three major steps. The first step is to map a corpus of texts into an incidence vector matrix (IVM). A corpus of texts means a collection of all texts. The IVM has n rows, one for each text, and m columns, one for each word. Each cell $C_{i,j}$ in IVM represents the frequency of a word i in a text j. The second step is to reduce the column dimension of IVM matrix by applying ordinary principal component analysis (PCA). PCA is a method that linearly transforms the incidence matrix into a new coordinate system, where the variation in the data can be described with fewer dimensions than the initial data. Thus, the second step transforms the original incidence vector matrix from $n \times m$ to an $n \times k$ matrix, where $k < m$, by choosing k components. The third step is to group the texts into different classes by utilizing clustering methods, such as K-means clustering (see Section 4.1 for details). However, due to the linearity of PCA, the reconstructed IVM matrix becomes a dense matrix with most non-zero entries. This results in a hard and time-consuming problem for clustering (see Section 3 for details). In this paper, I apply an advanced sparse principal component analysis (SPCA) to LSA. Unlike ordinary PCA, SPCA sets the less important features as zeros, accelerates the computations, and improves the interpretivity

of the reconstructed incidence vector matrix.

Most existing studies of latent semantic analysis focus on English and scarcely focus on Chinese because of the disparity between two languages. As a native speaker of Mandarin, I deal almost exclusively with texts in Chinese. Therefore, this paper focuses on LSA on Chinese to find out the issues specific to the language.

I decided to apply LSA to texts taken from the feeds users posted on Weibo, since Weibo is the biggest social media platform in China [12]. Many different topics are discussed on Weibo, such as sports, health and education. I chose to use the texts relevant to New York University (NYU), because NYU has always been one of the most popular schools among Chinese students. This year, NYU is even more famous on Weibo, since the trending topic was Taylor Swift's commencement speech this past May. Texts from Weibo can be extracted from the platform's official APIs freely. However, Weibo required complicated steps for verification and I did not have the permission to access it, so I had to use a third party source named Gooseeker (集搜客) to finish the extraction. The text data were downloaded as a csv file.

1.2 Paper structure

This paper is organized as follows: Section 2 explains the detail of data cleaning and data preparation for raw textual data. Section 3 shows two different ways to achieve LSA; one is LSA with principal component analysis and another is LSA with sparse principal component analysis. Section 4 gives results and comparisons. Finally, Section 5 describes the remaining challenges and the conclusion of this study.

2 | Data Cleaning

Cleaning up the text data is an essential step for text classification to prepare and highlight attributes. Cleaning text data consists of a number of steps, such as separating words, removing punctuation and stop words, and feature extraction. I needed to apply these steps to my sample data.

2.1 Data description

The textual data file I downloaded consists of 2,942 texts from Weibo that contains "NYU" and are dated between the 10^{th} of April to the 21^{st} of May. There are no missing observations in the sample data set. The first 5 rows are reported in Figure 1.

2.2 Word separation

Word separation for Chinese is very different than English. Chinese words are not delimited in the written language like English. In many cases, NLP researchers working with Chinese use an initial segmentation module that is intended to break a text into words. Additionally, texts from Weibo are composed of not only Chinese characters but also English words. Moreover, there are also many abbreviations and new slang words that are hard to be detected.

To successfully extract most of the words from our data, several check points are needed.

	userid	username	content	date	num_repost	num_comment	num_like
0	1233302995	李连源	【励志！#19岁听障男孩被世界名校录取#】日前，苏州19岁男孩张骏收到纽约大学的录取通知书。...	04月10日 23:46	1	评论	赞
1	5229300611	给我剥三斤荔枝	刷着刷着看见上海纽约大学 什么很多放弃top3的学生选择来这个学校 这么牛吗	04月10日 22:09	转发	3	1
2	2041405051	黑龙江卫视	【励志！#19岁听障男孩被世界名校录取#】日前，江苏苏州19岁男孩张骏收到纽约大学的录取通知...	04月10日 21:28	转发	评论	1
3	7575561018	美国申请小能手Magnolia	#美国留学答疑# 商科双非一本学校，有两段internships，现在GPA很低，只有2.8...	04月10日 21:25	转发	评论	赞
4	3243469590	成华残联	#自强如你# 【励志！#19岁听障男孩被世界名校录取#】日前，江苏苏州19岁男孩张骏收到纽约...	04月10日 20:24	转发	3	2

Figure 1: First 5 Rows from the Data Set.

The first check point is to locate all the English characters by checking the ASCII code of each character. This way, all the English words can be directly extracted from the first check point. The second checkpoint is to find the common combinations of words using Xinhua Dictionary APIs. The APIs are free to use and can request all Chinese words in Xinhua Dictionary. Specifically, I broke down each text into two-word combinations and matched them with 264,434 Chinese words from Xinhua Dictionary [10]. The last check point is to find the slang words as well as abbreviations, which is the most challenging step. Most of the Chinese slang words are not documented and they can even be combinations of Chinese characters and English characters, e.g. “早 C 晚 A” which is a slang word that describes someone who works hard and plays harder as he drinks coffee (“C”) in the morning (“早”) and alcohol (“A”) at night (“晚”).

To proceed with the task of the third check point, I chose to use an open-sourced word segmentation package called Jieba that contains a Chinese dictionary from Microsoft Research Asia and Peking University Institute. Jieba applies Viterbi algorithm for text recognition, created by Neuhoff et al, to spot the undocumented words [9]. Viterbi algorithm is a prediction dynamic algorithm that finds all the possible combination of words. Dynamic algorithm means that the result of the last iteration will be reused in the next iteration. Specifically,

	userid	username	content	date	num_repost	num_comment	num_like	separating_content
0	1233302995	李连源	【励志！#19岁听障男孩被世界名校录取#】日前，苏州19岁男孩张骏收到纽约大学的录取通知书。...	04月10日 23:46	1	评论	赞	[励志, 听障, 男孩, 世界, 名校, 录取, 日前, 苏州, 男孩, 张骏, 收到, 纽...
1	5229300611	给我剥三斤荔枝	刷着刷着看见上海纽约大学 什么很多放弃top3的学生选择来这个学校 这么牛吗	04月10日 22:09	转发	3	1	[看见, 上海, 纽约大学, 很多, 放弃, 学生, 选择, 这个, 学校]
2	2041405051	黑龙江卫视	【励志！#19岁听障男孩被世界名校录取#】日前，江苏苏州19岁男孩张骏收到纽约大学的录取通知...	04月10日 21:28	转发	评论	1	[励志, 听障, 男孩, 世界, 名校, 录取, 日前, 江苏, 苏州, 男孩, 张骏, 收...
3	7575561018	美国申请小能手Magnolia	#美国留学答疑# 商科双非一本学校，有两段internships，现在GPA很低，只有2.8...	04月10日 21:25	转发	评论	赞	[美国, 留学, 答疑, 商科, 双非, 一本, 学校, 两段, internships, ...
4	3243469590	成华残联	#自强如你# 【励志！#19岁听障男孩被世界名校录取#】日前，江苏苏州19岁男孩张骏收到纽约...	04月10日 20:24	转发	3	2	[自强, 励志, 听障, 男孩, 世界, 名校, 录取, 日前, 江苏, 苏州, 男孩, 张...

Figure 2: First 5 Rows from the Data Set with the "separated_content" column.

Viterbi algorithm treats the Chinese language as a Markov source and converts the words from the dictionary into a sequence of numbers; then the algorithm uses hidden Markov model to find the maximum likelihood of different combinations of Chinese characters using Bayes' theorem. Mathematically, this is the general formula for the possibility of each combination of words:

$$\mathrm{P}\left(z_0, \cdots, z_{i+1} \mid x_0, \cdots, x_{n+1}\right) \mathrm{P}\left(x_0, \cdots, x_{n+1}\right) = \prod_{i=1}^{n+1} \mathrm{P}\left(z_i \mid x_i\right) \mathrm{P}\left(x_i \mid x_{i-1}\right) \tag{2.1}$$

$\{x_0, ..., x_{n+1}\}$ represents all the combinations of Chinese characters in the dictionary; $\{z_0, ..., z_{t+1}\}$ is the set of t combinations of Chinese characters from observed data. Using the formula above, the slang words and abbreviations stand out with a specific probability, so Jieba helps to successfully conduct the third check point. After running the corpus of texts through check points, I obtained a list of words for each text, which can be seen in the column called "separated_content" in Figure 2.

2.3 Feature extraction

Feature extraction is the process of recognising and choosing important features from the data. The extraction can be done manually or automatically. In my case, I only needed the

column of "separated_content" from the original data set. I also needed to remove all the common insignificant words, identical texts, etc.

2.4 Removing insignificant words

Stop words, symbols, words highly relevant to NYU, and words that appear once are considered as insignificant words. Stop words are determiners, coordinating conjunctions, and prepositions. To filter out the stop words in my textual data, I used a list of stop words that Baidu (“百度”) once published. Symbols are emojis and punctuation marks. They have a specific range of ASCII codes, so I filtered them out by checking the ASCII code of each character. Words that are highly relevant to New York University, such as the name "NYU" in Chinese and English and its abbreviations, appear in almost all of the sample texts. Therefore, they are also not meaningful enough to take into consideration. Last but not least, words that only appear once cannot be clustered. Since I could not find their correlations with any other words, so I needed to remove them as well. To find the words that appear once, I created a table with one column, denoted “words” , of all the unique words that appeared in the data; and another column, denoted “count” , of the frequency corresponding to each word in Figure 3. The table shows that there are 11,273 words. After deleting the insignificant words, the total number of unique words is 5,041. Here is a snapshot of how the text changed in Table 2.1.

2.5 Removing bot-generated feeds

Removing identical feeds in the textual data is also necessary. I wrote a module with the reference from Twitter Botometer, created by Network Science Institute (IUNI) at Indiana University. If the texts contain over 10 words and appear to be highly identical over 2

	words	count
0	纽约大学	2876
1	荣誉	849
2	博士学位	808
3	霉霉	767
4	授予	710
...	...	...
11268	可怜	1
11269	长得	1
11270	几朵	1
11271	肝肠寸断	1
11272	初三	1

11273 rows × 2 columns

Figure 3: Table with all words and their frequencies.

Text	Original Text before	Text after removing insignificant words
Separating _content	【 励志 ! # 19 岁 听障 男孩 被 世界名校 录取 ...	励志 听障 男孩 世界名校 录取 ...

Table 2.1: One example of Removing Insignificant Words

times in the data set, then the similar texts will be marked as bot-generated feeds [15]. To improve the accuracy of the module, I used two similarity methods, cosine similarity (CS) and Spearman rank correlation coefficient (SRCC), to detect the similarities (see Subsection 2.5.1 for details). I only needed to keep the first appearance and removed the rest. After running the algorithm for all texts, I detected 338 repeated texts out of 2,941 feeds and removed all of them with 2,603 texts left. It shows that Weibo appears to have a serious problem of bot engagement and fake traffic, in which 11.5% of raw texts are bot-generated content.

2.5.1 Measures of similarity

There are two main approaches in similarity computation: the first is the Cosine Similarity, which exploits algebraic properties of vectors and their geometrical interpretation. Mathematically, Let u and v be the vector representations of two texts t_1 and t_2. Cosine similarity simply measures $\cos\theta$, where θ is the angle between u and v as follows:

$$\cos(\mathrm{u}, \mathrm{v}) = \frac{\mathrm{u} \cdot \mathrm{v}}{\|\mathrm{u}\|\|\mathrm{v}\|} \tag{2.2}$$

Second is Pearson's correlation coefficient (PCC). PCC shows the linear relationship between two variables, which constraints the diversity present in textual data [1]. To add the nonlinear properties on top of PCC, Spearman's rank correlation coefficient (SRCC) measures the strength and direction of association between two variables. Mathematically, the SRCC expression is shown below:

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} \left(d_{u_i, v_i}\right)^2}{n\left(n^2 - 1\right)} \tag{2.3}$$

Here, d means the difference in paired ranks and n is the number of dimensions for both text vectors. For example, I chose two text vectors from incidence matrix that contain the

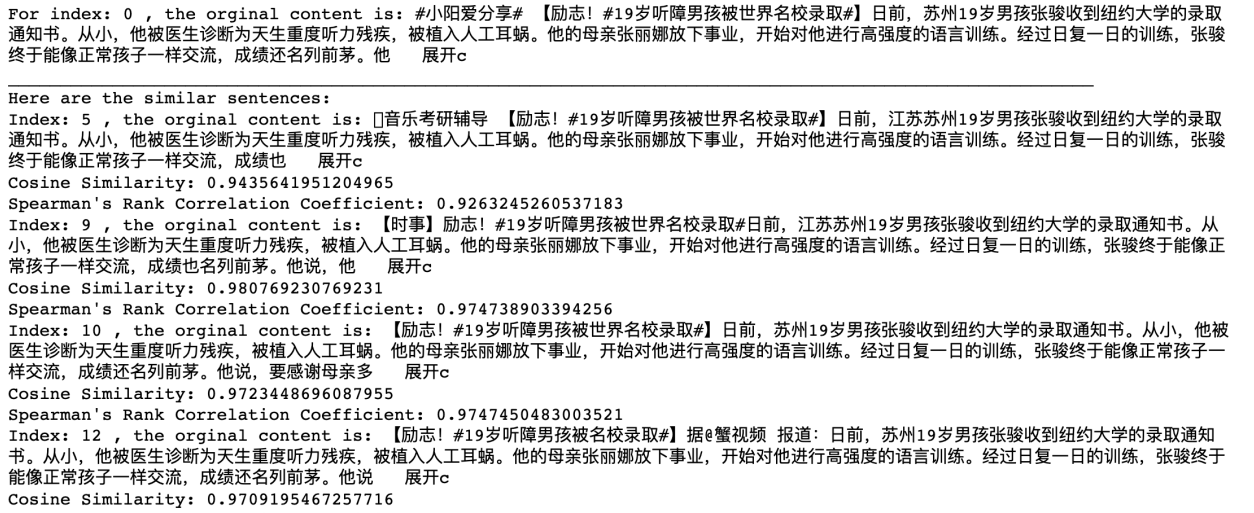

Figure 4: Example of Suspicious bot-generated feeds

similar topic in different wording that a 19 year old boy was accepted by NYU. I denoted the two texts as vectors u and v respectively. The similarity value of these two text vectors is shown below:

$\mathrm{CS}(u,v)$	$\mathrm{SRCC}(u,v)$
0.644646	0.913463

(2.4)

The results in (2.4) show that SRCC performs much better in finding the similarity with a similarity value of 91.3463%.

Therefore, I decided to combine the two methods to remove the suspicious bot-generated feeds. I set up the condition that if any two texts share the similarity percentage of CS and SRCC above a threshold of 0.90 and both texts contain over 10 characters, they are automatically marked as suspicious feeds. This is one example of detecting the bot-generated content in Figure 4.

3 | Latent Semantic Analysis with PCA and SPCA

LSA is a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of texts [6]. LSA converts the text data into a vector representation and uses the technique from PCA called singular value decomposition (SVD) to reduce the dimensions. After dimension reduction, the semantic space is established.

3.1 Vectorization of raw data

The first step of LSA requires a vector representation of the text data. I named the vector representation as incidence vector matrix (IVM). I created a module to generate such a IVM, where each row represents a unique text, each column represents a unique word, and each cell contains the frequency of a certain word in a text.

3.1.1 Applying TF-IDF vector technique

To emphasize the words with comparatively low frequencies, I applied the statistical measure called Term-frequency inverse document frequency method (TF-IDF). The TF-IDF method was introduced to add the weight for each cell [16]. Specifically, the TF-IDF technique

	0	1	2	3	4	5	...	5035	5036	5037	5038	5039	5040
word	纽约大学	荣誉	博士学位	霉霉	授予	taylorswift	...	迈阿密	梯度	当然	患者	混合	策略
Text1	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
Text2	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
Text4	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
Text6	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
Text2599	0	0	0	0	0	0.000172081	...	0	0	0	0	0	0
Text2600	4.83491e-05	0.000153463	0	0.000162214	0	0	...	0	0	0	0	0	0
Text2601	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
Text2602	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0
Text2603	4.83491e-05	0	0	0	0	0	...	0	0	0	0	0	0

2376 rows × 5041 columns

Figure 5: IVM with TF-IDF Filtration

updates each cell in IVM with a new weight, which indicates how important a word is to a text. Specifically, $tf_{i,j}$ = number of occurrences of a word i in a text j divided by the total number of words in the text j, df_i = number of texts containing i, N = total number of texts, and $w_{i,j}$ = the certain weight for a word i in a text j with a expression as follows:

$$\mathrm{w}_{i,j} = \mathrm{tf}_{ij} \times \log \frac{\mathrm{N}}{\mathrm{df}_i} \tag{3.1}$$

Basically, after applying TF-IDF filtration, the rarer the word is, the higher $w_{i,j}$ is going to be. I stored that vectorised data in M_tfidf in Figure 6. Here is the module I wrote for the TF-IDF vectoriser in Figure 7.

3.2 Usage of principal component analysis

Principal component analysis in LSA is used for dimension reduction. PCA can be computed via singular value decomposition (SVD) of the original data. In detail, PCA decomposes the normalized incidence vector matrix into the product of three matrices: the first component

```
def get_tfidf_matrix(M,train_df):
    M_tfidf = M.copy()
    for i in range(len(M)):
        for j in np.where(M[i]!=0):
            tf = M[i][j]/M[:,j].sum()
            df = M[i].sum()
            N = M.shape[1]
            w = tf *math.log10(N/df)
            M_tfidf[i][j]=w
    return M_tfidf
```

Figure 6: Alogrithm for TF-IDF Technique

matrix, denoted U, describes the original row entities (texts) as vectors of derived orthogonal factor values; another, denoted V, describes the original column (words) entities in the same way. The third matrix called Σ is a diagonal matrix containing single values. As the mathematical expression shows, SVD of incidence vector matrix X is $X = U\Sigma V^T$, where X is an $n \times m$ matrix of the normalized observed data (IVM), U is an $n \times n$ unitary matrix, Σ is an $n \times m$ rectangular diagonal matrix, and V^T is an $m \times m$ unitary matrix, which is the transpose of V. $U\Sigma$ are the principal components, and the columns of V are the corresponding loadings of the principal components. Since principal components sequentially capture the maximum variability among the columns of X, minimal information loss is guaranteed. The first couple $k(k \ll \min(n, m))$ PCs often capture most of the variance, so the PCs will be chosen to represent the data, thus a dimensionality reduction is achieved.

In other words, PCA can also be viewed as a projection that minimizes the total squared reconstruction error. Specifically, the projection can be written as $\min \|X - \bar{X}\|_F^2$, where X is the IVM, and $\bar{X}$ is the reconstructed IVM.

The reconstruction error can be computed from the method as follows. $\bar{X}$ can be computed using rank-k approximation. The result of $\min \|X - \bar{X}\|_F^2$ is the top-k singular values of X that minimizes the Frobenious norm of the difference with the matrix X. Recalling that

the SVD of X, $\sigma_k u_k$ is the k-th PC, the PCA formulation using the low-rank approximation can be derived as:

$$(d, u, v) = \arg\min_{u,v} \left\| X - duv^T \right\|_F^2 \quad \text{s.t. } \|u\| = 1 \tag{3.2}$$

In our study, the incidence vector matrix was $X = [x_1, x_2, \ldots, x_{4443}]^T \in R^{2422\times4443}$. By following the expression above, I set up the error term to be $1e-16$, step size to be 0.5, and chose 2,000 as the starting point to approximate the result. I eventually reduced the features to $2,336$ from the approximation module. However, to capture over 80% of variance of all sample data, I still needed 694 features. Each PC is a linear combination of all 694 non zero features, which is very difficult to interpret.

3.3 Usage of sparse principal component analysis

Sparse principal component analysis (SPCA) achieves the dimension reduction as well as a better interpretability. The concept of SPCA was introduced in the seminal work in 1996 by Tibshirani, where sparsity constraints were enforced on the singular values [3]. Specifically, SPCA method adds a l_1 penalty, aka Lasso constraint [13], that is directly integrated into the regression criterion, leading to a modified PCA with sparse weights. Here is the formula to demonstrate:

$$\begin{gathered}(u_{new}, \beta) = \arg\min_{u,\beta} \frac{1}{2}\|U_i\Sigma_{ii} - X\beta\|_{\mathrm{F}}^2 + l_1\|\beta\|_1 \\ \text{subject to } \|u_i\|_2 <= 1 \text{ for all } 0 \le k < m_{\text{components}}\end{gathered} \tag{3.3}$$

u_{new} stand for new unitary matrices for ith row of U and V in SVD, U_i, Σ_{ii} are the original ith row of U, Σ from the SVD, and $v_{new} = \frac{\beta}{\|\beta\|} = V_i$, which is the ith column of V in SVD . The sparsity-inducing $\|\cdot\|_1$ matrix norm prevents the loading, v, from noise. The degree

of penalization can be adjusted through the iteration times and the constraint l_1. Smaller values lead to a gently regularized factorization, while larger values shrink many coefficients to zeros, thus adding the sparsity.

However, the original SPCA lasso method suffers several limitations. For example, if there are high-dimensional data (n) with few observations (m), the LASSO SPCA method selects at most m variables before it saturates. Also if there is a group of highly correlated variables, then the LASSO SPCA method tends to select one variable from a group and ignore the others. To overcome these limitations, Zou and Hastie proposed a new method for SPCA, called the elastic net in 2006 [18]. The elastic net adds a quadratic part ($\|\beta\|^2$) to the penalty, which was used in ridge regression. The elastic net formulates the sparse PCA problem as the following ridge regression problem plus a LASSO penalty. The formula is shown below:

$$(\beta) = \arg\min_{\beta} \|U_i\Sigma_{ii} - \mathrm{X}\beta\|^2 + \lambda\|\beta\|^2 + \lambda_1\|\beta\|_1, \tag{3.4}$$

Here the indeterminacy from lasso SPCA method is eliminated by the positive ridge penalty ($\|\beta\|^2$). The coefficients after normalization are independent of λ, therefore the ridge penalty is to ensure the reconstruction of principal components. The larger λ is, the higher the simplicity of the reconstruction of principal components will be.

In this thesis, I chose to apply the elastic net method and find the parse loadings via an alternating manifold proximal gradient method (A-ManPG), which was created by Zou, et al in 2020 [2]. The A-ManPG algorithm uses the following subproblems with an alternating updating scheme to solve SPCA, computed in a Gauss-Seidel manner for faster convergence, which means an iterative method used to solve a system of linear equations.

$$D_k^A := \arg\min_{D^A} \left\langle \nabla_A H\left(A_k, B_k\right), D^A \right\rangle + f\left(A^k + D^A\right) + \tfrac{1}{2t_1}\left\|D^A\right\|_F^2 \text{ s.t. } D^A \in \mathrm{T}_{A_k}\mathcal{M}_1 \tag{3.5}$$

$$D_k^B := \arg\min_{D^B} \left\langle \nabla_A H\left(A_{k+1}, B_k\right), D^B \right\rangle + f\left(B^k + D^B\right) + \frac{1}{2t_2}\left\|D^B\right\|_F^2 \text{ s.t. } D^B \in \mathrm{T}_{B_k}\mathcal{M}_2 \tag{3.6}$$

where $H(A,B) = \mathrm{Tr}\left(B^T X^T X B\right) - 2\,\mathrm{Tr}\left(A^T X^T X B\right)$, $A = U\Sigma$, $B = V$, k is the number of desired principal component, $t_1, t_2 > 0$ are step sizes, f(A) $\equiv$ 0,$\mathrm{M}_1 = \mathrm{St}(p,k)$, where $\mathrm{St}(p,r) := \{A \in \mathbb{R}^{p\times r} : A^\top A = I_r\}$ is the Stiefel manifold, and $\mathcal{M}_2 = \mathbb{R}^{p\times k}$. The sub-problems are solved using an adaptive semi-smooth Newton method. Here is the pseudo-code for A-ManPG method:

Algorithm 1 A-ManPG Method

Ensure: Initial point (A_0, B_0), parameter $\gamma \in (0,1)$, step sizes $t_1 \le 1/L_A$ and $t_2 \le 1/L_B$.
  for $i = 0, 1, ..., n,$ do
    Solve the A-subproblem in (3.5) to obtain D_k^A.
    Set $\alpha_1^k = 1$
    while $F\left(\mathrm{Retr}_{A_k}\left(\alpha_1^k D_k^A\right), B_k\right) > F\left(A_k, B_k\right) - \frac{\alpha_1^k}{2t_1}\left\|D_k^A\right\|_F^2$ do
      $\alpha_1^k = \gamma\alpha_1^k$
    end while
    Set $A_{k+1} = \mathrm{Retr}_{A_k}\left(\alpha_1^k D_k^A\right)$
    Solve the B-subproblem in (3.6) to obtain D_k^B.
    Set $\alpha_2^k = 1$
    while $F\left(A_{k+1}, \mathrm{Retr}_{B_k}\left(\alpha_2^k D_k^B\right)\right) > F\left(A_{k+1}, B_k\right) - \frac{\alpha_2^k}{2t_2}\left\|D_k^B\right\|_F^2$ do
      $\alpha_2^k = \gamma\alpha_2^k$
    end while
    $B_{k+1} = \mathrm{Retr}_{B_k}\left(\alpha_2^k D_k^B\right)$
  end for

3.4 Classification using K-means clustering method

Clustering is the final step of latent semantic analysis. Clustering means to group the vectors of texts that are similar together. I applied K-means clustering method to the reconstructed incidence vector matrix. K-means clustering method is a numerical, unsu-

pervised, non-deterministic iterative method common for clustering large sets of data [7]. This clustering method takes the number of clusters from users and assigns labels to the nearby n-dimensional data points based on minimizing the squared Euclidean distance as minimizing the dissimilarity between a data point and a cluster. The simple mathematical expression is shown below:

$$\sum_{i=0}^{n} \min_{\mu_j \in C} \left(\left\| x_i - \mu_j \right\|^2 \right) \tag{3.7}$$

where x_i is the i-th data point and mu_j is the j-th centroid. So I applied K-means to the reconstructed IVM from PCA and SPCA, respectively.

4 | Evaluation of the Empirical Results

The results of this study consist of two parts: first is the empirical results from K-means clustering result of the reconstructed incidence vector matrix from PCA and SPCA. Second is emphasizing on checking the interpretability of SPCA compared to PCA using supervised machine learning methods (support vector machine and naïve Bayes).

4.1 Empirical results of SPCA and PCA from K-means

I applied both PCA and SPCA to the reconstructed incidence vector matrix before doing k-means clustering. I also took the run time into consideration, which enables me to track and understand how long the k-means method takes for each data set to be clustered. I first set the principal component to be 685, which contained 80% of variance, and SPCA the number of cluster to be 5 to see how both incidence vector matrices will perform if I want to extract 5 trendy topics from the sample data. However, the clustering performances for both PCA are very poor. K-means algorithm clustered 2,387 texts for one cluster and a total of 35 texts in other clusters. Then, I calculated the within cluster variance for cluster from 5 to 200. Eventually I successfully identified the elbow point as 30 and the number of principal components to 685 to expose the trendy topics.

Table 4.1 shows the empirical results, which are the top 5 clusters out of 30 clusters, from the result of K-means method on PCA and SPCA, respectively. The table presents

dr-ml method	Clusters	Number of texts	Related term	Run time (sec)
PCA km	Cluster 1	1037	泰勒 (Taylor Swift)	0.182
	Cluster 2	917	毕业 (Graduation)	
	Cluster 3	261	申请 (Application)	
	Cluster 4	80	租房 (Renting)	
	Cluster 5	24	刘畅 AK(Liu Chang)	
SPCA km	Cluster 1	1051	泰勒 (Taylor Swift)	0.158
	Cluster 2	936	毕业 (Graduation)	
	Cluster 3	237	申请 (Application)	
	Cluster 4	78	租房 (Renting)	
	Cluster 5	20	研究 (Research)	

k-means is abbreviated by km and I picked the best result from different combination of the λ_1 and λ_2 for SPCA, which are 0.01,0.1,0.001 and 1,5,10,15, respectively.

Table 4.1: The empirical performance of SPCA and PCA from K-means clustering.

the amount of texts per cluster, related words, and run time for K-means method. From the results, SPCA did a better job than PCA on capturing more texts relevant to the top two trendy topics, while both gave a good picture of the trendy topics relevant to NYU. These topics include Taylor Swift, graduation, application, a Chinese rapper called "AK" graduating from NYU, renting, and research. In addition, K-means method with SPCA is faster than PCA; the faster computation can be interpreted as SPCA suppressed the less important features to be zero and the euclidean distance between texts vectors on different topics would be much wider, so the k-means clustering method took less time to compare the texts with centroids. Besides, SPCA also helped find another cluster that talked about researches relevant to NYU professors.

I note that SPCA did not group the texts that are relevant to Chinese rapper AK together, compared to PCA. One possible reason for this could be due to the diverse choice of words people used to talk about the rapper and the high penalty of λ_2, causing more zero weights for the features; however, this problem requires further research on linguistics.

4.2 Further experiment on SPCA and PCA

To better understand the difference between SPCA and PCA, I also applied the following experiment to test the interpretability with naïve Bayes method and support vector machine method. I chose the top six trendy topics (Taylor Swift, Offering, Chinese Rapper, Application, and renting) shown in the results of unsupervised K-means clustering on PCA and SPCA. I manually checked each cluster and labeled the texts that are relevant to the trendy topics. There are 1,052 texts relevant to Taylor Swift, 47 texts of the Chinese rapper AK, 160 texts of application, 141 texts of renting, and 296 texts of offers. I created a new incidence vector matrix with only the texts that have been labeled, and applied PCA and SPCA again to the new incidence vector matrix. Then, to prevent selection bias, I applied cross validation to two separate IVMs and fed them into naïve Bayes method and support vector machine method. The result is revealed in Section 4.2.3.

4.2.1 Evaluation metrics

I used three primary evaluation metrics defined below:

$$\text{Accuracy} = \frac{\#\text{ of right predictions}}{\text{total }\#\text{ of observations}}$$

$$\text{Precision} = \frac{\#\text{ of right predictions}}{\text{total }\#\text{ of predictions}}$$

$$\text{Recall} = \frac{\#\text{ of right prediction in one label}}{\text{total }\#\text{ of number of texts that are actually within this label}}.$$

In addition to these three evaluation metrics, I also considered another two criteria. which are the fit time and predict time.

4.2.2 Supervised learning classifier methods

Ensemble learning methods refer to the techniques that create multiple models and then combine them to produce better results than any of the single models individually. I utilised two ensemble learning methods for my experiment.

4.2.2.1 Naïve Bayes

Naive Bayes algorithm can be defined as a supervised classification algorithm which is based on Bayes theorem with an assumption of independence among features. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple.

4.2.2.2 Support vector machine

Support Vector Machine (SVM) is another supervised learning algorithm, which is primarily used for classification problems in Machine Learning. SVM builds a model that assigns new examples to one category or the other. The goal of the SVM algorithm is to create a line called hyperplane that can segregate n-dimensional space into classes so that people can easily put the new data point in the correct category in the future. The further distance between data points and the hyperplane, the better classification result will be.

4.2.3 Empirical results with supervised learning classifier methods

I applied both PCA and SPCA to my new incidence vector matrix before implementing the two supervised learning classifiers naïve Bayes and support vector machine.

Table 4.2 presents the empirical results on accuracy, precision and recall, as well as the

dr-ml method	Number of components	Accuracy	Precision	Recall	Fit time (sec)	Prediction time (sec)
PCA svm	2	0.862	0.830	0.993	0.767	0.210
SPCA svm λ_1=0.01, λ_2=20	2	0.755	0.775	0.965	0.079	0.016
PCA svm	4	0.853	0.821	0.993	0.652	0.223
SPCA svm λ_1=0.01, λ_2=10	4	0.828	0.8	0.982	0.028	0.007
PCA svm	6	0.846	0.814	0.993	0.836	0.347
SPCA svm λ_1=0.001, λ_2=20	6	0.821	0.792	0.986	0.036	0.011
PCA svm	8	0.839	0.807	0.993	0.826	0.275
SPCA svm λ_1=0.01, λ_2=20	8	0.812	0.781	0.989	0.028	0.010
PCA nb	2	0.927	0.926	0.965	0.078	0.059
SPCA nb λ_1=0.1, λ_2=5	2	0.720	0.914	0.632	0.001	0.0002
PCA nb	4	0.933	0.924	0.979	0.076	0.024
SPCA nb λ_1=0.001, λ_2=10	4	0.789	0.911	0.751	0.001	0.0003
PCA nb	6	0.929	0.913	0.986	0.068	0.032
SPCA nb λ_1=0.01, λ_2=10	6	0.745	0.903	0.684	0.001	0.0002
PCA nb	8	0.931	0.916	0.986	0.063	0.024
SPCA nb λ_1=0.1, λ_2=20	8	0.734	0.889	0.677	0.001	0.0002

Also, naive Bayes and Support vector machine are abbreviated by nb and svm, respectively

Table 4.2: The empirical performance of SPCA and PCA from svm and nb classification.

fit time and prediction time for the SPCA and PCA dimensionality reduction and machine learning methods, compared to naïve Bayes method. The results also suggest that SPCA performs overall as good as PCA in terms of accuracy and precision using support vector mechine method, while it tends to be generally much faster to train and predict our sample data set compared to PCA. The reason of such result can be seen as the text vectors in sparse IVM from SPCA being well separated. For the bad performance of NB of SPCA, the reason is highly relevant to the correlation among principal components, but one of the assumptions of NB is that all the features are independent. Witten et al. [14] provided an answer for this correlation issue. The faster computations of SPCA in both classifiers imply that the interpretability is indeed desirable. The interpretability can also be seen that the number of components used is important too. The accuracy and precision as well as the recall of both SPCA and PCA decrease when the number of principal components increased, probably because more noisy information is used with more components.

Unlike the accuracy and precision performances, SPCA did not perform well in terms of the recall performance, compared to PCA. One possible reason for this is that many weights for loadings were suppressed to be zeros, however this problem still requires further research.

5 | Conclusion

In this thesis, I studied the text classification performance of sparse principal component analysis compared to principal component analysis in Chinese language. I applied sparse PCA and ordinary PCA to the texts on Chinese social media platform Weibo. The analysis required working with a high dimensional sparse matrix obtained from such text data, which I used SPCA to handle and analyse the sparse matrix. My empirical results showed that SPCA did perform as well as PCA in the k-means unsupervised learning method, except the difference for the last cluster, as K-means with PCA produced the cluster for Chinese rapper but the other produced the cluster for researcher. How does this difference happen? I believe that this is due to the sparsity of SPCA that converted the frequency of less common features to be zeros, while PCA still preserved the non-zeros features. Moreover, from research by Zidani, Chinese social media users are inclined to praise or comment on social media in diverse words and subversive expression amid heavy censorship- they will be creating new slang words or using abbreviations to comment on their idols [17].

What's more, to test the interpretability of SPCA, I also tried another experiment with labeled texts. The empirical results showed that SPCA is not as accurate as PCA in terms of accuracy, precision, and recall. However, all the evaluation metrics were very close to PCA in support vector machine method, whilst SPCA showed two main advantages over PCA. First, SPCA resulted in sparse principal components so the left IVM was much easier to interpret compared to PCA. Second, the computational time is much faster for SPCA.

Faster calculation is important especially for big data situations.

SPCA did not perform well in terms of the recall performance for naïve Bayes method. This is a topic that needs further linguistic research for natural language processing using SPCA. Overall, the move from PCA to SPCA on Chinese language is advantageous, as discussed above. The raw text data that converted into sparse matrix showed the need for dimensionality reduction as a feature selection process for selecting the components used in the machine learning classifiers.

Finally, there are other problems to be studied when using SPCA. For example, the correlation between sparse principal components caused the naïve Bayes to be less accurate. For ordinary PCA, the principal components are expected to be orthogonal as the features are orthogonal; however, this expectation does not fundamentally hold for sparse components. In SPCA, the loadings were forced to be zero because of the lasso penalty, so the uncorrelated property was sacrificed. Thus, a good choice of classification method is also necessary when using SPCA.

Bibliography

[1] Nino Arsov et al. A measure of similarity in textual data using Spearman's rank correlation coefficient. Nov. 2019. url: https://arxiv.org/abs/1911.11750v1.

[2] Shixiang Chen et al. "An alternating manifold proximal gradient method for sparse principal component analysis and sparse canonical correlation analysis". In: INFORMS Journal on Optimization 2.3 (2020), pp. 192–208. doi: 10.1287/ijoo.2019.0032.

[3] Alexandre d'Aspremont, Francis Bach, and Laurent El Ghaoui. Optimal Solutions for Sparse Principal Component Analysis. 2007. doi: 10.48550/ARXIV.0707.0705. url: https://arxiv.org/abs/0707.0705.

[4] Scott Deerwester et al. "Indexing by latent semantic analysis". In: Journal of the American Society for Information Science 41.6 (1990), pp. 391–407. doi: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9. url: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9.

[5] Ammar Ismael Kadhim. "Survey on supervised machine learning techniques for automatic text classification". In: Artificial Intelligence Review 52.1 (2019), pp. 273–292. doi: 10.1007/s10462-018-09677-1.

[6] Thomas K. Landauer and Susan T. Dumais. "A solution to Plato's problem: The Latent Semantic Analysis Theory of acquisition, induction, and representation of knowledge."

In: Psychological Review 104.2 (1997), pp. 211–240. doi: 10.1037/0033-295x.104.2.211.

[7] J MacQueen. "Classification and analysis of multivariate observations". In: 5th Berkeley Symp. Math. Statist. Probability. 1967, pp. 281–297.

[8] Fred Morstatter et al. Measuring the impact of Isis Social Media Strategy - Stanford University. May 2018. url: https://snap.stanford.edu/mis2/files/MIS2_paper_23.pdf.

[9] David L. Neuhoff. "The Viterbi algorithm as an aid in text recognition (Corresp.)" In: IEEE Trans. Inf. Theory 21 (1975), pp. 222–226.

[10] Pwxcoo. Pwxcoo/Chinese-xinhua: 中华新华字典数据库。包括歇后语，成语，词语，汉字。. url: https://github.com/pwxcoo/chinese-xinhua.

[11] C. SaranyaJothi and D.Thenmozhi D.Thenmozhi. "Machine learning approach to document classification using concept based features". In: International Journal of Computer Applications 118.20 (2015), pp. 33–36. doi: 10.5120/20864-3578.

[12] Lai Lin Thomala. Weibo corporation: Daus 2022. June 2022. url: https://www.statista.com/statistics/1058070/china-sina-weibo-dau/#:~:text=As%20of%20the%20first%20quarter,582%20million%20monthly%20active%20users.

[13] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: Journal of the Royal Statistical Society: Series B (Methodological) 58.1 (1996), pp. 267–288. doi: 10.1111/j.2517-6161.1996.tb02080.x.

[14] D. M. Witten, R. Tibshirani, and T. Hastie. "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis". In: Biostatistics 10.3 (2009), pp. 515–534. doi: 10.1093/biostatistics/kxp008.

[15] Kai-Cheng Yang, Emilio Ferrara, and Filippo Menczer. Botometer 101: Social bot practicum for computational social scientists. 2022. doi: 10.48550/ARXIV.2201.01608. url: https://arxiv.org/abs/2201.01608.

[16] Wen Zhang, Taketoshi Yoshida, and Xijin Tang. "A comparative study of TF*IDF, LSI and multi-words for text classification". In: Expert Systems with Applications 38.3 (2011), pp. 2758–2765. doi: 10.1016/j.eswa.2010.08.066.

[17] Sulafa Zidani. "Represented Dreams: Subversive Expressions in Chinese Social Media as Alternative Symbolic Infrastructures". In: Social Media + Society 4.4 (2018), p. 2056305118809512. doi: 10.1177/2056305118809512. eprint: https://doi.org/10.1177/2056305118809512. url: https://doi.org/10.1177/2056305118809512.

[18] Hui Zou, Trevor Hastie, and Robert Tibshirani. "Sparse principal component analysis". In: Journal of Computational and Graphical Statistics 15.2 (2006), pp. 265–286. doi: 10.1198/106186006x113430.