Principles of Data Science Project 1 Dimensionality Reduction

Junhao Dai daijunhao@sjtu.edu.cn

April 5, 2024

1 Introduction

1.1 Experimental Configuration

The Python version used for this experiment is 3.11.5, with the following library versions:

- NumPy version: 1.26.2
- Pandas version: 2.0.3
- Scikit-learn version: 1.3.0

Additionally, it's worth noting that this experiment was conducted without the use of servers; all programs were run on a laptop with a CPU model AMD Ryzen 7 5800H with Radeon Graphics, operating at 3.20 GHz.

1.2 My Work

In this project, we are required to explore the optimal dimensionality reduction method and the optimal dimensionality to comlpete the task of data classification.Firstly we download Animals with Attributes (AwA2) dataset from https://cvml.ist.ac.at/AwA2/.

First, we read the files AwA2-features.txt and AwA2-labels.txt from the downloaded dataset. The former contains feature files that have been pre-extracted using deep learning methods, while the latter contains label information. We divide the dataset into training and testing sets, with the testing set comprising 40% of the data. Next, we use k-fold cross-validation on the training set to determine the hyperparameter C (penalty parameter) for the SVM model.

We use the SVM model with the best C parameter as our baseline. Then, we employ feature selection, feature projection, and feature learning methods to reduce the dimensionality of the training set. We obtain the SVM model's prediction accuracy under different methods and dimensions as evaluation criteria to explore the optimal dimensionality reduction method and dimension.

2 Experiments and Results

In this section, I will provide a detailed description of my experiment's methodology and corresponding results. For the baseline, we utilized the SVM model with the optimal C parameter determined through k-fold cross-validation.

In the feature selection step, we employed two methods: SelectKBest and Variance Threshold. For feature projection, we utilized the kernelPCA method. Lastly, in feature learning, we employed the t-SNE method and LLE method. Since the principles of each method have been thoroughly understood through the course, I won't delve into redundant explanations here.

2.1 Baseline

K-fold cross-validation is a technique used to evaluate the performance of machine learning models. In this method, the original dataset is randomly divided into K subsets, called "folds." Then, the model is trained K times, each time using K-1 folds of data to train the model and the remaining one fold to evaluate the model's performance. In this experiment, we set the parameter k (i.e., the number of folds) to 5.

Table 1: Results of coarse grid search for optimal C parameter

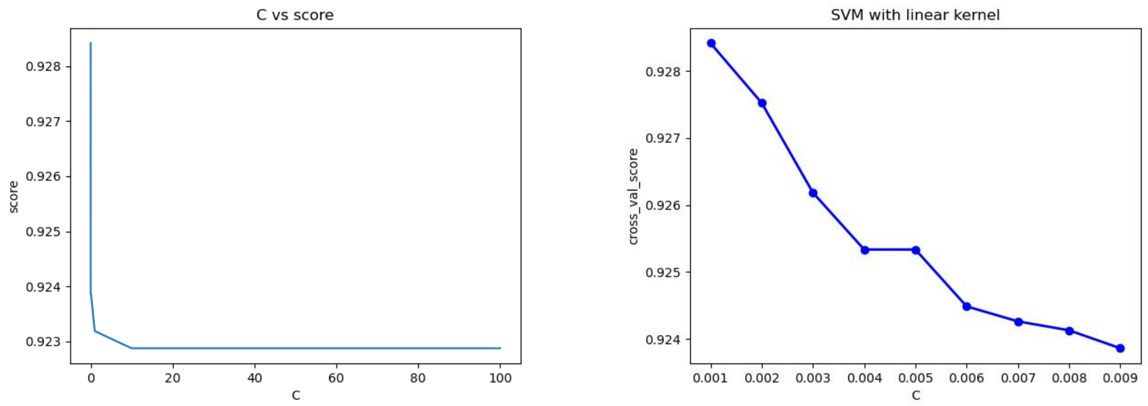
C	**cross_val_score**
0.001	0.928415
0.01	0.923860
0.1	0.923860
1	0.923190
10	0.922878
100	0.922878

Initially, we searched for the optimal C parameter within a coarse range, and the results are presented in Table 1.It is evident that the optimal C parameter should be around 0.01. Therefore, we narrowed down the range from 0.01 to 0.1 with a step size of 0.02, obtaining fine-grained results as shown in Table2. The results of the two rounds of k-fold cross-validation are illustrated in Figure 1 and Figure 2. Based on the above results, we set the optimal C parameter to be 0.01.

Table 2: Results of fine-grained grid search for optimal C parameter

C	cross_val_score
0.001	0.928415
0.002	0.927522
0.003	0.926182
0.004	0.925333
0.005	0.925333
0.006	0.924485
0.007	0.924262
0.008	0.924128
0.009	0.923860

The accuracy obtained by SVM using the optimal hyperparameter C is as shown in Table 3. Subsequently, we will use this as our baseline.



(a) Select K Best

(b) Varianve Threshold

Fig 1: The results of coarse-grained and fine-grained searches for the optimal C parameter.

Table 3: Results of fine-grained grid search for optimal C parameter

Best C	Best Acc(%)
0.001	93.2078

2.2 Feature Selection

2.2.1 Variance Threshold

The Variance Threshold method is a commonly used technique in feature selection, aimed at filtering out features with variances below a certain threshold. We employ different variance threshold values [0.1, 0.2, 0.3, 0.5, 0.7, 0.9] to perform dimensionality reduction on the data. Similarly, we use the accuracy of the SVM model as the utility evaluation criterion.

Table 4: Results of Variance Threshold Method

n_comp	accuracy	dimension
0.10	0.929131	1883
0.20	0.930337	1353
0.30	0.929868	1024
0.50	0.926519	623
0.70	0.921897	400
0.90	0.915266	268

Table 4 displays the dimensions and prediction accuracies obtained under different variance threshold values for dimensionality reduction. The optimal result is achieved with a variance threshold of 0.2, yielding a dimensionality of 0.53 and an accuracy of approximately 93.03%. Similarly to SelectKBest, it shows a slightly lower accuracy compared to the baseline. This may be due to the same reason: dimensionality reduction leading to a slight loss in precision.

2.2.2 Select-K-Best

SelectKBest is a commonly used method in feature selection, aiming to select the K most relevant features from the dataset with respect to the target variable. This method evaluates the correlation between each feature and the target variable based on statistical tests and selects the top K features with the highest scores. Common scoring functions include the chi-squared test and the F-test (ANOVA). In this experiment, ANOVA F-value is selected. K is set to [2, 5, 10, 20, 50, 100, 200, 500, 750, 1000, 1200, 1500, 2000]. The utility evaluation criterion in this experiment is the accuracy of the SVM model after dimensionality reduction.

Table 5: Results of Select K Best Method

n_comp	accuracy
2.00	0.118695
5.00	0.216826
10.00	0.308192
20.00	0.500904
50.00	0.775805
100.00	0.861143
200.00	0.898855
500.00	0.922500
750.00	0.925849
1000.00	0.927859
1200.00	0.928528
1500.00	0.930404
2000.00	0.929131

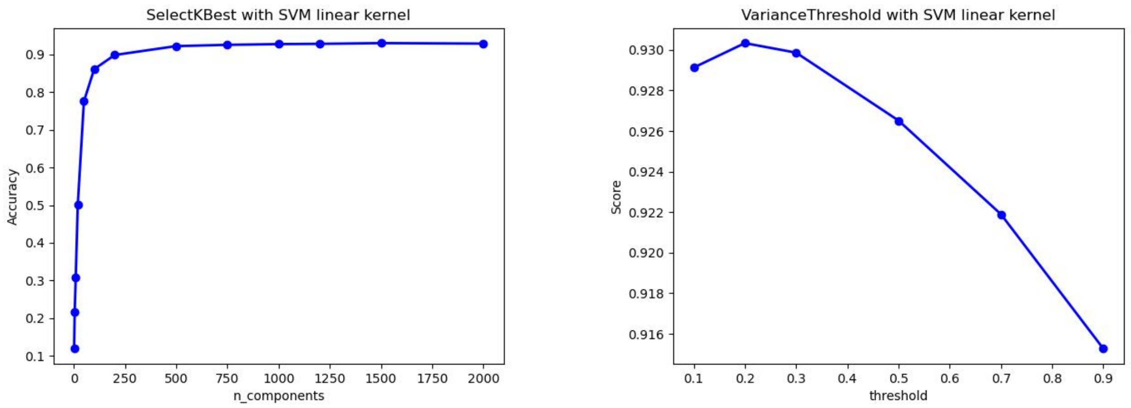
From Table5, it is evident that the optimal dimensionality is around 1500 dimensions, achieving the highest accuracy of approximately 93.04%. Compared to the baseline, the accuracy is slightly lower, which could be attributed to the slight loss of precision caused by dimensionality reduction.The relationship between dimensions and accuracy for the two methods of feature selection is illustrated in Figure.2.

2.3 Feature Projection

2.3.1 kernel PCA

PCA (Principal Component Analysis) is a commonly used unsupervised dimensionality reduction technique aimed at reducing the number of features while preserving the variance of the dataset. It achieves this by linearly transforming the original feature space into a new feature space. Kernel PCA (Kernel Principal Component Analysis), on the other hand, is an extension of PCA that introduces nonlinear mappings into PCA, allowing for dimensionality reduction on nonlinear problems. In this section, we utilize Kernel PCA with the objective of achieving target dimensions set as [2, 5, 10, 20, 50, 100, 200, 500, 750, 1000, 1200, 1500, 2000]. Similarly, we use the accuracy of the SVM model after dimensionality reduction as the utility evaluation criterion.

The result is shown in Figure3 and Table 6 it is evident that the best performance is achieved when reducing dimensions to 500 using kernel PCA, with an SVM model accuracy of approximately



(a) coarse-grained results

(b) fine-grained results

Fig 2: The results of Select K Best and Varianve Threshold

92.78%. Similarly to previous methods, this accuracy is slightly lower than the baseline. One possible reason could be that the reduction of components in PCA eliminates some effective features.

Table 6: Results of kernel PCA Method

n_comp	accuracy
2.00	0.105834
5.00	0.350124
10.00	0.631657
20.00	0.818541
50.00	0.888874
100.00	0.910041
200.00	0.920289
500.00	0.927792
750.00	0.923572
1000.00	0.922098
1200.00	0.920088
1500.00	0.919753
2000.00	0.915667

2.4 Feature Learning

2.4.1 tSNE

t-SNE[1] (t-distributed Stochastic Neighbor Embedding) is a technique used to map high-dimensional data into two- or three-dimensional space while preserving both local and global similarities between the original data points as much as possible. The main idea behind t-SNE is to represent each data point in high-dimensional space as a point in low-dimensional space, such that similar data points are close to each other in the low-dimensional space while dissimilar data points are far apart. In this section, Barnes-Hut approximation is utilized to compute gradients in two and three dimensions. The perplexity range is set as [10.0, 20.0, 30.0, 40.0, 50.0]. Once again, the accuracy of the SVM model after dimensionality reduction serves as the utility evaluation criterion.

Table 7: Accuracy for different perplexity values and linear components

Perplexity	Linear Components	Accuracy
10.0	2.00	0.086610
	3.00	0.017550
20.0	2.00	0.086945
	3.00	0.014870
30.0	2.00	0.087012
	3.00	0.018822
40.0	2.00	0.086811
	3.00	0.019492
50.0	2.00	0.086409
	3.00	0.015071

From Table 7, when perplexity is set to 30.0, t-SNE performs the best in two dimensions, but the accuracy is only 8.7%. Therefore, I believe that the t-SNE method reduces the dimensionality too much, resulting in insufficient data to extract effective features. The result is shown in Figure 4

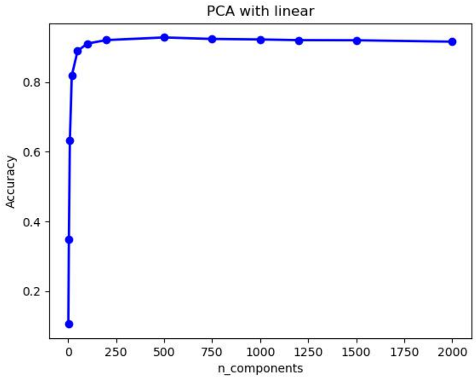


Fig 3: Accuracy for Different n_copms in PCA Method

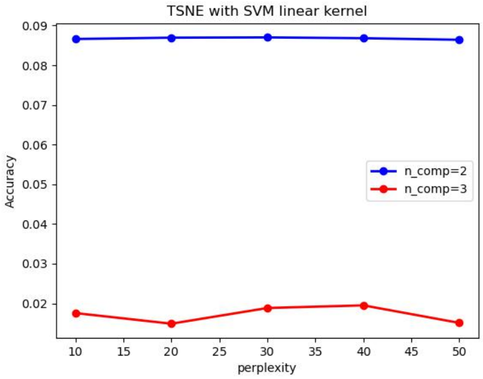


Fig 4: Accuracy for different perplexity values and linear components

2.4.2 LLE

LLE[2] (Locally Linear Embedding) is a nonlinear dimensionality reduction technique used to map high-dimensional data into a lower-dimensional space while preserving local linear relationships between data points as much as possible. LLE achieves this by reconstructing the linear relationships between data points within local neighborhoods, thereby preserving the local structure of the data.

In this section, we set the number of neighbors to 4, 8, and 16, and the target dimensionality range to [2, 3, 50, 100, 500, 1000, 2000]. Table **??** displays the experimental results. We can observe that retaining more neighbors and dimensions contributes to improving the accuracy of the SVM model, and the best result is achieved when the number of neighbors is set to 16 and the dimensionality is set to 2000, with an SVM model accuracy of 90.93% shown in Figure 5.

Table 8: Accuracy for different neighbor values and linear components

Neighbor	Linear Components	Accuracy
4	2.00	0.092438
	3.00	0.097595
	50.00	0.808828
	100.00	0.862951
	500.00	0.890348
	1000.00	0.895037
	2000.00	0.895438
8	2.00	0.091098
	3.00	0.122714
	50.00	0.832474
	100.00	0.874205
	500.00	0.900730
	1000.00	0.904481
	2000.00	0.904012
16	2.00	0.122848
	3.00	0.154531
	50.00	0.833478
	100.00	0.871793
	500.00	0.904012
	1000.00	0.907495
	2000.00	0.909304

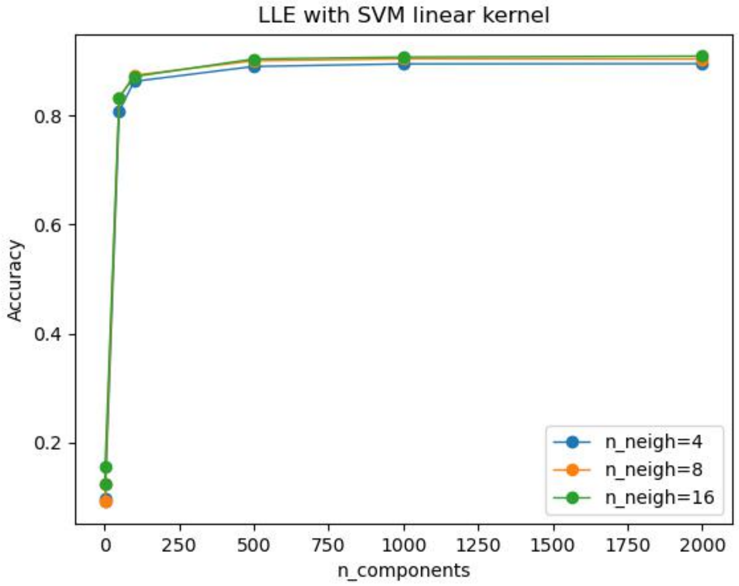


Fig 5: Accuracy for different neighbor values and linear components

3 Conclusion

In this experiment, we approached data dimensionality reduction for classification models from three perspectives: Feature Selection, Feature Projection, and Feature Learning. From the results, it appears that the selection methods, specifically Select-K-Best and VarianceThreshold, achieved the highest model accuracy. Feature Learning methods, on the other hand, such as t-SNE and LLE, had slower training times (t-SNE took approximately two hours, while LLE took around eight hours) and obtained lower accuracy compared to the former two.

The reason for this could be speculated as follows: the richness of the existing dataset might be insufficient to capture enough nonlinear data relationships. Consequently, Feature Learning methods might not adequately extract effective features due to the lack of nonlinear data relationships, resulting in lower accuracy compared to the simpler selection methods, which minimize the loss of effective information.

参考文献

[1] Jersson X Leon-Medina, Maribel Anaya, Francesc Pozo, and Diego Tibaduiza. Nonlinear feature extraction through manifold learning in an electronic tongue classification task. *Sensors*, 20(17):4834, 2020.

[2] Rongzhen Zhao, Kunju Shi, Zhaohui Li, and Tao Zhang. F-lle algorithm and its application in fault feature extraction. In *2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 418–422, 2013.