Homework2

Machine Learning

1 PCA Algorithm

The conventional PCA algorithm (Eigen-decomposition or SVD based), as shown in Alg. 1 and Alg. 2, is based on a linear projection of the data onto a subspace of lower dimensionality than the original data space. However, we can also define nonlinear principal component models in the original data space through Kernel PCA, as shown in Alg. 3.

1.1 Conventional PCA Algorithm

Algorithm 1: Eigen-decomposition based PCA.

```
Input  : $n$ points $(x_1, x_2, \dots, x_n)$ in $D$-dim
Output: The first principal component $w$
1 Initialize $X$ $(N \times D)$ and the $i$-th row of $X$ is $(x_i - \bar{x})^T$;
2 Carry out Eigen-decomposition for $X^T X$:
      $$X^T X \leftarrow U \Sigma U^T;$$
3 Since the covariance matrix is $N^{-1} X^T X$, we have $w = u_1$;
4 return $w$;
```

Algorithm 2: SVD-based PCA.

```
Input  : $n$ points $(x_1, x_2, \dots, x_n)$ in $D$-dim
Output: The first principal component $w$
1 Initialize $X$ $(N \times D)$ and the $i$-th row of $X$ is $(x_i - \bar{x})^T$;
2 Carry out SVD for $X$, where $V = (v_1, v_2, \dots, v_D)$:
      $$X \leftarrow U \Sigma V^T;$$
3 Since $X^T X \leftarrow V \Sigma^2 V^T$ (the covariance matrix is $N^{-1} X^T X$), we have $w = v_1$;
4 return $w$;
```

Advantages of conventional PCA algorithms:

- Conventional PCA is easy to implement.
- Compared with Eigen-decomposition based method, SVD-based PCA can be computed with some fast methods.
- With U and V, SVD-based PCA can compress data not only by reducing the redundant features (with V) but also by removing redundant samples (with U). In general, it can compress data in two directions.

Limitations:

- They are not able to express non-linear principal components.
- The computation cost may increase largely as data scale up.

1.2 Kernel PCA Algorithm

```
Algorithm 3: Kernel PCA.
   Input: $n$ points $(x_1, x_2, \ldots, x_n)$ in $D$-dim, the kernel function
 1 Compute the covariance matrix in the feature space:
       $$C \leftarrow XX^T,$$
     where $X = [\phi(x_1), ..., \phi(x_N)]$;
 2 The eigenvector $v_i$ of $C$ can be written in the form:
       $$v_i \leftarrow \sum_{n=1}^{N} a_{in}\phi(x_n);$$
 3 Solve $\mathbf{a}_i$ with kernel function and compute the first principal component $v_m$.
 4 The projection of a point $x$ onto vector $v_m$ can also be represented in terms of kernel function:
       $$y_m(x) = \sum_{n=1}^{N} a_{mn}k(x, x_n);$$
```

Kernel PCA can reveal non-linear information of the datasets. With mapping the data points to a higher-dimensional space, it may find out some more information of the dataset. However, this method can also be of large cost with the computation of kernel matrix. Moreover, the choice of kernel function may affect the final performance of this method, which in turn makes this method less robust.

2 Factor Analysis (FA)

According to the Bayes Rule:

$$\begin{aligned} p(y|x) &= \frac{p(x|y)p(y)}{p(x)} \\ &= \frac{G(x|Ay+\mu, \Sigma_e)G(y|0, \Sigma_y)}{p(x)} \end{aligned}$$

Since $x = Ay + \mu + e$, we have:

$$E[x] = E[Ay + \mu + e] = \mu,$$

and

$$\begin{aligned} \Sigma_x &= E[(x - E[x])(x - E[x])^T] \\ &= E[Ayy^TA^T + ey^TA^T + Aye^T + ee^T] \\ &= A\Sigma_yA^T + \Sigma_e \end{aligned}$$

Therefore, we have:

$$
\begin{aligned}
p(y|x) &= \frac{p(x|y)p(y)}{p(x)} \\
&= \frac{G(x|Ay+\mu,\Sigma_e)G(y|0,\Sigma_y)}{p(x)} \\
&= \frac{G(x|Ay+\mu,\Sigma_e)G(y|0,\Sigma_y)}{G(x|\mu,A\Sigma_y A^T+\Sigma_e)}
\end{aligned}
$$

then $\mu_{z|x} = \Sigma_y A^T(A\Sigma_y A^T + \Sigma_e)^{-1}(x-\mu)$ and $\Sigma_{z|x} = \Sigma_y - \Sigma_y A^T(A\Sigma_y A^T + \Sigma_e)^{-1}A\Sigma_y^T$

3 Independent Component Analysis (ICA)

Suppose the mixing matrix A is orthogonal, and source signal s_i has a Guassian distribution. Then the distribution of (x_1, x_2) $(x = As)$ is completely symmetric Guassian and contains no information on the directions of columns of A. Therefore any A will result in the same distribution of (x_1, x_2) and A cannot be estimated. So we have to maximize the non-Gaussianity of s_i, which can be done though finding w that maximize the non-Gaussianity of $y = w^T x = z^T s$, according to Central Limit Theorem.

4 Dimensionality Reduction by FA

In this section, the performances of BIC and AIC on model selection are evaluated in terms of sample size (N), dimensionality of y (m), noise level (σ^2) and dimensionality of data (n).

4.1 Sample Size

In this part we evaluate the performance of BIC and AIC with different sample sizes. The datasets for evaluation are randomly generated data points with $m = 3$, $n = 10$, $\sigma^2 = 0.1$ and $\mu = 0$. The mixing matrix A is randomly generated. The results are shown in Fig 1. As shown in Fig. 1, the performance of BIC and AIC are almost the same when the sample size is large, while AIC performs better when the sample size is small. Since BIC takes into consideration the sample size, it is likely that when the sample size si small the $\frac{d_m}{2}\ln N$ plays a more dominant role in the BIC computation.

4.2 Dimensionality of y

In this part we evaluate the performance of BIC and AIC with different dimensionality of y. The datasets for evaluation are randomly generated data points with $N = 500$, $n = 10$, $\sigma^2 = 0.1$ and $\mu = 0$. The mixing matrix A is randomly generated. The results are shown in Fig 2. As shown in Fig. 2, the performance of AIC improves as the gap between the dimensionality of data and the dimensionality of y decreases. Moreover, when the sample size is large enough, BIC has a relatively stable performance compared with AIC in term of different dimensionality of y.

4.3 Dimensionality of data

In this part we evaluate the performance of BIC and AIC with different dimensionality of data. The datasets for evaluation are randomly generated data points with $m = 3$, $N = 2000$, $\sigma^2 = 0.1$ and $\mu = 0$. The mixing matrix A is randomly generated. The results are shown in Fig 3. As shown in Fig. 3, the performance of BIC and AIC are almost the same, and there is no significant influence of dimensionality

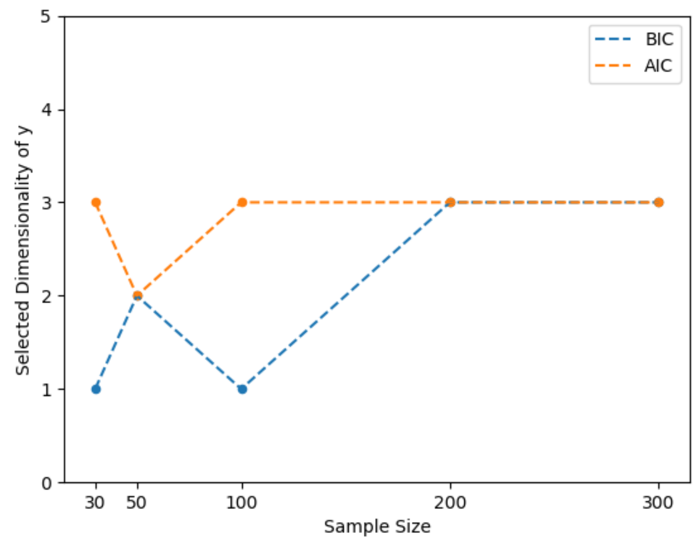

Figure 1: The model selection performances of BIC and AIC in terms of different sample sizes.

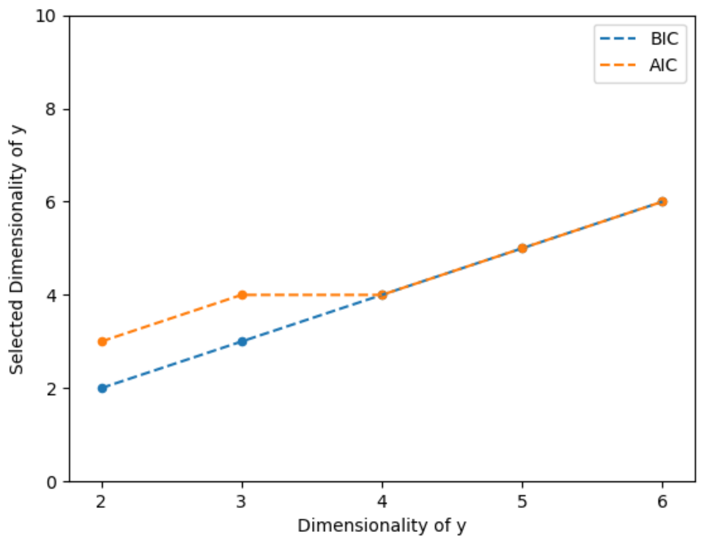

Figure 2: The model selection performances of BIC and AIC in terms of different dimensionality of y.

of data on the model performance. BIC has a relatively stable performance compared with AIC as the gap between the dimensionality of y and the dimensionality of data increases.

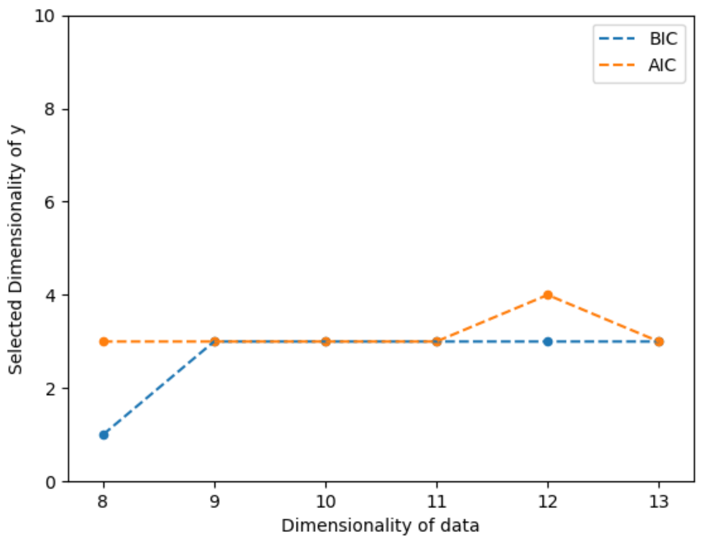

Figure 3: The model selection performances of BIC and AIC in terms of different dimensionality of data.

4.4 Noise Level

In this part we evaluate the performance of BIC and AIC with different noise level. The datasets for evaluation are randomly generated data points with $m = 3$, $n = 10$, $N = 2000$ and $\mu = 0$. The mixing matrix A is randomly generated. The results are shown in Fig 4. As shown in Fig. 4, the performance of both methods become worse when the noise level increases, which is easy to understand. Moreover, it seems that AIC can be more tolerant to the increase of noise level.

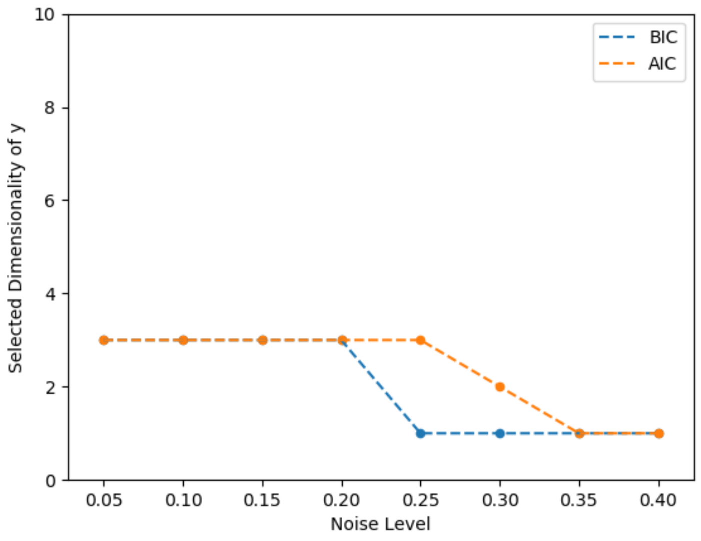

Figure 4: The model selection performances of BIC and AIC in terms of different sample sizes.

5 Spectral Clustering

In this experiment, we generate several different kinds of dataset, run popular clustering algorithms on them and analyze when spectral clustering works well and when it would fail. All the datasets are 2-dimension with the sample size of 1500. Tab. 1 shows a detailed information about the datasets. The evaluation result is shown in fig. 5. As we can see, spectral clustering works extremely well when the classification is non-linear (such as dataset #1 and #2), when the dataset is with anisotropic distribution (dataset #4) and when the centers of different clusters are relatively far from each other (dataset #5).

Table 1: Details of the Datasets.

Datasets	Generated Function	Parameters
1	sklearn.datasets.make_circle()	factor=0.5, noise=0.05
2	sklearn.datasets.make_moons()	noise=0.05
3	sklearn.datasets.make_blobs()	with varied variance
4	sklearn.datasets.make_blobs()	with transformation matrix
5	sklearn.datasets.make_blobs()	noise=0.05
6	np.random.rand	2D

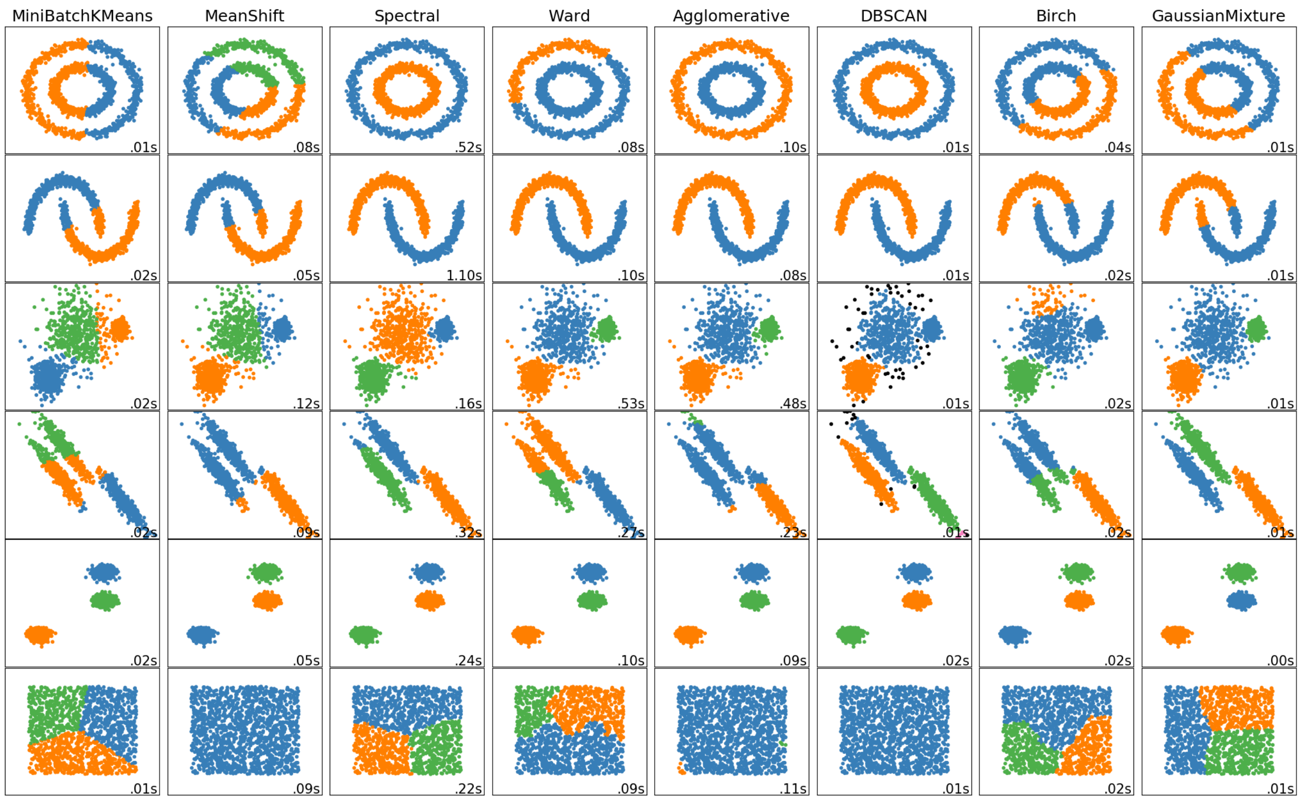

Figure 5: Performance of different clustering algorithms on 6 different types of datasets.

However, the performance of spectral clustering is relatively poor when the data of different clusters have no significant features (dataset #6) and the centers of different clusters are close to each other. Also, if we do not set the initial cluster number for spectral clustering, it well have a poor performance compared with other algorithms.