Machine Learning Homework 2*

Hongzhou Liu
517030910214
deanlhz@sjtu.edu.cn

1 PCA algorithm

1.1 Eigenvalue Decomposition

The original PCA adopts eigenvalue decomposition as a solution to find principal components.

Algorithm 1: PCA based on Eigenvalue Decomposition

Input : Dataset $\mathbf{X} = \{\mathbf{x_1}, \cdots, \mathbf{x}_N\}$ where $\mathbf{x_i} \in \mathbb{N}^d$
Output : The first principal component $\mathbf{w}$

1 Normalize $\mathbf{X}$ to make sure the mean is $\mathbf{0}$

2 Calculate the covariance matrix of $\mathbf{X}$ as

$$\mathbf{\Sigma} = \mathbf{X}\mathbf{X}^T$$

3 Calculate the eigenvalues and eigenvectors of $\mathbf{\Sigma}$

4 Choose the maximum eigenvalue λ_1 and the corresponding eigenvector $\mathbf{x}_1$

5 Calculate the first principal component

$$\mathbf{w} = \mathbf{x}_1^T\mathbf{X}$$

6 **return** $\mathbf{w}$

Advantages

- Quite easy to understand and easy to implement

Disadvantages

- When $\mathbf{X}$ is of high dimension, the computation of $\mathbf{X}\mathbf{X}^T$ is expensive
- The eigenvalue decomposition is not so efficient and computation expensive in high dimensions
- It's hard to interpret the meaning of principal components found by the algorithm

1.2 Singular Value Decomposition

SVD is another approach of matrix decomposition. It can be also used to find principal components. The SVD is like

$$X_{m\times n} = U_{m\times m}\Sigma_{m\times n}V^T_{n\times n} \tag{1}$$

where U and V are orthogonal matrices and Σ contains singular values on it's diagonal. If $\mathbf{X}$ is our dataset, then $\mathbf{U}$ is actually made up of eigenvectors of $\mathbf{X}\mathbf{X}^T$, the covariance matrix.

*GitHub repo: https://github.com/DeanAlkene/CS420-MachineLearning/tree/master/A2

Algorithm 2: PCA based on Singular Value Decomposition

Input : Dataset $\mathbf{X} = \{\mathbf{x_1}, \cdots, \mathbf{x}_N\}$ where $\mathbf{x_i} \in \mathbb{N}^d$
Output : The first principal component $\mathbf{w}$

1 Normalize $\mathbf{X}$ to make sure the mean is $\mathbf{0}$

2 Apply SVD on $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

3 Multiply $\mathbf{U}^T$ on the both side and denote it as $\mathbf{X}'$

$$\mathbf{U}^T\mathbf{X} = \mathbf{\Sigma}\mathbf{V}^T = \mathbf{X}'$$

4 Let the first row of $\mathbf{X}'$ be $\mathbf{w}$

5 **return** $\mathbf{w}$

Advantages

- There is iterative methods to solve SVD and we don't need to calculate $\mathbf{X}\mathbf{X}^T$ it will be more efficient than doing eigenvalue decomposition
- SVD can reduce dimension in both row and column directions, while eigenvalue decomposition cannot
- SVD can solve non-square matrices while eigenvalue decomposition cannot

Disadvantages

- The sparsity of data might be lost
- It's also hard to interpret the meaning of decomposed matrices found by the algorithm

2 Factor Analysis (FA)

By Bayesian formula, we know that

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{y})p(\mathbf{y})}{p(\mathbf{x})} \tag{2}$$

Here,

$$p(\mathbf{x}) = p(\mathbf{A}\mathbf{y} + \mu + \mathbf{e}) \tag{3}$$

and

$$p(\mathbf{x}|\mathbf{y}) = G(\mathbf{x}|\mathbf{A}\mathbf{y} + \mu, \Sigma_e), p(\mathbf{y}) = G(\mathbf{y}|0, \Sigma_y) \tag{4}$$

generally

$$p(\mathbf{e}) = G(\mathbf{e}|\mu_e, \Sigma_e) \tag{5}$$

Here, $\mathbf{A}\mathbf{y} + \mu$ is an affine transformation of $\mathbf{y}$, thus

$$p(\mathbf{x}) = G(\mathbf{A}\mathbf{y} + \mu|\mu, \mathbf{A}\Sigma_y\mathbf{A}^T) + G(\mathbf{e}|\mu_e, \Sigma_e) = G(\mathbf{x}|\mu + \mu_e, \mathbf{A}\Sigma_y\mathbf{A}^T + \Sigma_e) \tag{6}$$

Then,

$$p(\mathbf{y}|\mathbf{x}) = \frac{G(\mathbf{x}|\mathbf{A}\mathbf{y} + \mu, \Sigma_e)G(\mathbf{y}|0, \Sigma_y)}{G(\mathbf{x}|\mu + \mu_e, \mathbf{A}\Sigma_y\mathbf{A}^T + \Sigma_e)} \tag{7}$$

The density function of Gaussian distribution is

$$G(\mathbf{x}|\mu, \mathbf{\Sigma}) = \frac{1}{\sqrt{(2\pi)^k|\mathbf{\Sigma}|}} \exp(-\frac{1}{2}(\mathbf{x} - \mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x} - \mu)) \tag{8}$$

k is the dimension of $\mathbf{x}$. Then we consider the exponential terms of $p(\mathbf{y}|\mathbf{x})$ which is

$$-\frac{1}{2}(\mathbf{x}-\mathbf{A}\mathbf{y}-\mu)^T\Sigma_e^{-1}(\mathbf{x}-\mathbf{A}\mathbf{y}-\mu)-\frac{1}{2}\mathbf{y}^T\Sigma_y^{-1}\mathbf{y}+\frac{1}{2}(\mathbf{x}-\mu+\mu_e)^T(\mathbf{A}\Sigma_y\mathbf{A}^T+\Sigma_e)^{-1}(\mathbf{x}-\mu+\mu_e) \tag{9}$$

We only consider terms containing $\mathbf{y}$, that is

$$\begin{aligned} &-\frac{1}{2}[-\mathbf{x}^T\Sigma_e^{-1}\mathbf{A}\mathbf{y} - \mathbf{y}^T\mathbf{A}^T\Sigma_e^{-1}(\mathbf{x}-\mathbf{A}\mathbf{y}-\mu) + \mu^T\Sigma_e^{-1}\mathbf{A}\mathbf{y} + \mathbf{y}^T\Sigma_y^{-1}\mathbf{y}] \\ &= -\frac{1}{2}[(\mu-\mathbf{x})^T\Sigma_e^{-1}\mathbf{A}\mathbf{y} + \mathbf{y}^T\mathbf{A}^T\Sigma_e^{-1}(\mu-\mathbf{x}) + \mathbf{y}^T(\mathbf{A}^T\Sigma_e^{-1}\mathbf{A} + \Sigma_y^{-1})\mathbf{y}] \end{aligned} \tag{10}$$

We know that

$$\begin{aligned} &(\mathbf{y}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{y}-\mu) \\ &= \mathbf{y}^T\Sigma^{-1}\mathbf{y} - \mathbf{y}^T\Sigma^{-1}\mu - \mu^T\Sigma^{-1}\mathbf{y} + \mu^T\Sigma^{-1}\mu \end{aligned} \tag{11}$$

Compare 10 and 11 we get,

$$\Sigma_{\mathbf{y}|\mathbf{x}} = (\mathbf{A}^T\Sigma_e^{-1}\mathbf{A} + \Sigma_y^{-1})^{-1} \tag{12}$$

and

$$\Sigma_{\mathbf{y}|\mathbf{x}}^{-1}\mu_{\mathbf{y}|\mathbf{x}} = \mathbf{A}^T\Sigma_e^{-1}(\mathbf{x}-\mu) \tag{13}$$

Hence

$$p(\mathbf{y}|\mathbf{x}) = G(\mathbf{y}|(\mathbf{A}^T\Sigma_e^{-1}\mathbf{A} + \Sigma_y^{-1})^{-1}\mathbf{A}^T\Sigma_e^{-1}(\mathbf{x}-\mu), (\mathbf{A}^T\Sigma_e^{-1}\mathbf{A} + \Sigma_y^{-1})^{-1}) \tag{14}$$

3 Independent Component Analysis (ICA)

In ICA, we have a linear combination of source vectors $\mathbf{x} = \mathbf{A}\mathbf{s}$ where $\mathbf{s}$ are independent sources. The goal is to find a transformation $\mathbf{W}$ to seperate each sources into $\mathbf{y}$ and make each entry in $\mathbf{y}$ as independent as possible.

The Central Limit Theorem tells us that a sum of independent random variables from arbitrary distributions tends torwards a Gaussian distribution, under certain conditions. Let's consider ICA as

$$\mathbf{y} = \mathbf{w}^T\mathbf{x} = \mathbf{w}^T\mathbf{A}\mathbf{s} = (\mathbf{w}^T\mathbf{A})\mathbf{s} = \mathbf{z}^T\mathbf{s} \tag{15}$$

Now, $\mathbf{y}$ is a liner combination of random variables $\mathbf{s}$. According to the Central Limit Theorem, $\mathbf{y}$ should be closer to Gaussian than any s_i in $\mathbf{s}$. However, to pursue independence among each entry of $\mathbf{y}$, we ought to minimize affects of being closer to Gaussian brought by $\mathbf{z}^T$. It is equally to say, we should take $\mathbf{w}$ that maximizes the non-Gaussianity, which is a principal for ICA estimation.

In another perspective, let's prove that in ICA at most one Gaussian variable is allowed. Let's consider $\mathbf{x} = \mathbf{A}\mathbf{s}$ where $\mathbf{s} = s_1, s_2$. Without lossing of generality, let $\mathbf{s} \sim \mathcal{N}(0, I)$. Then,

$$\mathbf{x} \sim \mathcal{N}(0, \mathbf{A}\mathbf{A}^T) \tag{16}$$

Here is an orthogonal transformation matrix $\mathbf{R}$. Apply it on $\mathbf{A}$ as $\mathbf{A}' = \mathbf{A}\mathbf{R}$, we have

$$\mathbf{x}' = \mathbf{A}\mathbf{R}\mathbf{s} \sim \mathcal{N}(0, \mathbf{A}\mathbf{R}\mathbf{R}^T\mathbf{A}^T) = \mathcal{N}(0, \mathbf{A}\mathbf{A}^T) \tag{17}$$

Thus, due to the symetric property of multivariable Gaussian, we cannot tell the source $\mathbf{s}$ from the observation $\mathbf{x}$ because there're infinite much $\mathbf{s}$. In this way, we also proved that, to implement ICA we should stay away from Gaussian.

4 Dimension Reduction by FA

In this section, I generated datasets based on following Factor Analysis model and did experiments on different settings of parameters of FA. Then I ran AIC and BIC model selection on those datasets and compared the performances of them.

$$\mathbf{x} = \mathbf{A}\mathbf{y} + \mu + \mathbf{e} \tag{18}$$

$$p(\mathbf{x}|\mathbf{y}) = G(\mathbf{x}|\mathbf{A}\mathbf{y} + \mu, \sigma^2\mathbf{I}) \tag{19}$$

$$p(\mathbf{y}) = G(\mathbf{y}|0, \mathbf{I}) \tag{20}$$

4.1 Experiment on sample size N

In this experiment, I fixed $n = 10$, $m = 3$, $\mu = 0$, $\sigma^2 = 0.1$ and tested AIC and BIC model selection on $N = \{50, 100, 200, 500, 1000, 2000, 5000\}$. As seen in 1, the performances of both AIC and BIC fluctuate when the sample size N increases. However, BIC will always choose a solution not worse than AIC. And in most of times, BIC finds a solution closer to the optimal, which is 3. The fluctuation may due to the randomized matrix A when generating datasets.

Table 1: Experiment on sample size N

N	n	m	μ	σ^2	m^*_{AIC}	m^*_{BIC}	$J_{AIC}(m^*_{AIC})$	$J_{BIC}(m^*_{BIC})$
50					6	4	-493.601677	-588.246816
100					6	5	-982.810455	-1111.766380
200					5	5	-1661.446367	-1824.713076
500	10	3	0	0.1	5	4	-4379.957981	-4588.581082
1000					6	4	-8168.984016	-8411.917903
2000					6	4	-15455.789912	-15733.034584
5000					5	4	-42272.641493	-42595.242556

4.2 Experiment on observed variable dimension n

In this experiment, I fixed $N = 500$, $m = 3$, $\mu = 0$, $\sigma^2 = 0.1$ and tested AIC and BIC with n in range of $\{2, 3, 5, 8, 10, 15, 20, 50, 100\}$. As we can see in 2, when n increases, both m^*_{AIC} and m^*_{BIC} also increase monotonically. In this case, the performances of AIC and BIC are quite the same when n is small or large. But BIC will perform better when $n \in [5, 20]$. It also shows that when n becomes larger, the performances of both AIC and BIC are not so good.

Table 2: Experiment on dimension n

N	n	m	μ	σ^2	m^*_{AIC}	m^*_{BIC}	$J_{AIC}(m^*_{AIC})$	$J_{BIC}(m^*_{BIC})$
	2				1	1	-1396.354846	-1604.977946
	3				1	1	-1718.675231	-1927.298332
	5				3	3	-2621.684229	-2830.307329
	8				5	3	-3638.028832	-3846.651933
500	10	3	0	0.1	5	4	-4655.394542	-4864.017643
	15				10	8	-5445.357437	-5653.980538
	20				13	10	-6640.627083	-6849.250184
	50				40	40	-11009.236851	-11217.859952
	100				90	90	-14643.727527	-14852.350627

4.3 Experiment on latent variable dimension m

In this experiment, I fixed $N = 500$, $n = 10$, $\mu = 0$, $\sigma^2 = 0.1$ and ran AIC and BIC with m in range of $\{1, 2, 3, 5, 8, 10, 15, 20, 50\}$. As usual, we can see 3 that BIC chooses a number smaller or equal than AIC and thus gets closer to the optimal. But when $m > n$ in which we are trying to increase dimension, the number that AIC and BIC choosed will be limited below 10, which is the value of n.

4.4 Experiment on the mean μ

In this experiment, I fixed $N = 500$, $n = 10$, $m = 10$, $\sigma^2 = 0.1$ and tested AIC and BIC with μ over $\{-2.0, -1.0, -0.8, -0.5, -0.2, 0, 0.2, 0.5, 0.8, 1.0, 2.0\}$. The result also shows that 4, though fluctuation occurs, BIC will perform better than AIC in most cases. In some cases, BIC hits the optimal value of 3.

Table 3: Experiment on dimension m

N	n	m	μ	σ^2	m^*_{AIC}	m^*_{BIC}	$J_{AIC}(m^*_{AIC})$	$J_{BIC}(m^*_{BIC})$
		1			5	3	-2714.345491	-2922.968592
		2			6	4	-3221.695130	-3430.318231
		3			6	4	-4504.367828	-4712.990928
		5			6	5	-5266.175263	-5474.798364
500	10	8	0	0.1	7	7	-6538.478659	-6747.101760
		10			7	7	-7201.846576	-7410.469677
		15			8	7	-8157.844961	-8366.468062
		20			7	7	-8959.720000	-9168.343101
		50			6	6	-11266.871473	-11475.494574

Table 4: Experiment on the mean μ

N	n	m	μ	σ^2	m^*_{AIC}	m^*_{BIC}	$J_{AIC}(m^*_{AIC})$	$J_{BIC}(m^*_{BIC})$
			-2.0		5	4	-4495.860128	-4704.483228
			-1.0		6	5	-4469.707856	-4678.330957
			-0.8		5	4	-4264.262261	-4472.885361
			-0.5		5	4	-4042.243784	-4250.866885
			-0.2		5	3	-4303.592699	-4512.215800
500	10	3	0	0.1	6	4	-4310.182025	-4518.805126
			0.2		6	3	-4213.409350	-4422.032451
			0.5		6	4	-4315.162901	-4523.786002
			0.8		6	4	-4003.357713	-4211.980814
			1.0		5	4	-4129.851244	-4338.474345
			2.0		5	4	-4769.434959	-4978.058060

4.5 Experiment on noise level σ^2

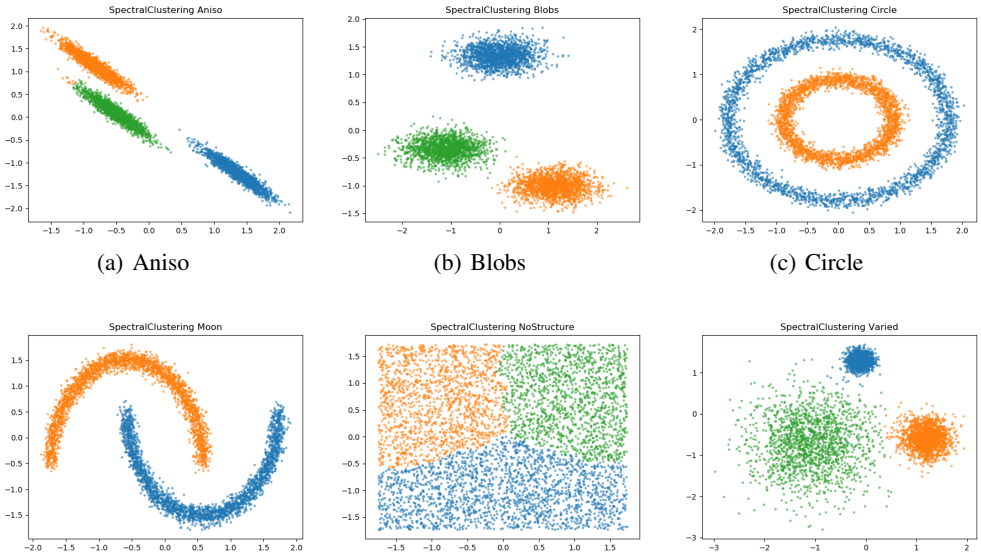
In this experiment, I fixed $N = 500$, $n = 10$, $m = 10$, $\mu = 0$ and tested AIC and BIC on $\sigma^2 = \{0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0\}$. It 5 again proves that BIC performs better than AIC in model selection in most cases. But the result fluctuates strangely, it may reveal that as the noise level increases, the tendency of uncertainty of the results occurs. And in this case, BIC is more stable than AIC.

Table 5: Experiment on noise level σ^2

N	n	m	μ	σ^2	m^*_{AIC}	m^*_{BIC}	$J_{AIC}(m^*_{AIC})$	$J_{BIC}(m^*_{BIC})$
				0.0001	5	3	-3322.894166	-3531.517267
				0.001	6	4	-2728.148545	-2936.771645
				0.01	5	4	-2287.788435	-2496.411536
500	10	3	0	0.1	5	4	-4312.271311	-4520.894412
				1.0	6	4	-8103.451885	-8312.074986
				10.0	8	2	-13098.779997	-13307.403098
				100.0	4	4	-18672.091688	-18880.714789
				1000.0	8	3	-24448.748153	-24657.371254

5 Spectral clustering

I compared Spectral Clustering and Gaussian Mixture Model on different datasets provided by `sklearn`. As we can see 5 Spectral Clustering works quite well on Blobs, Circle, Moon and Varied datasets, but no so well on Aniso and No Structure datasets. It shows that Spectral Clustering is suitable for data in special distributions. And in `sklearn`, the similarity matrix is based on distances between data points. The performance of Spectral Clustering always affeted by the metric of similarity and the construction of similarity graph.

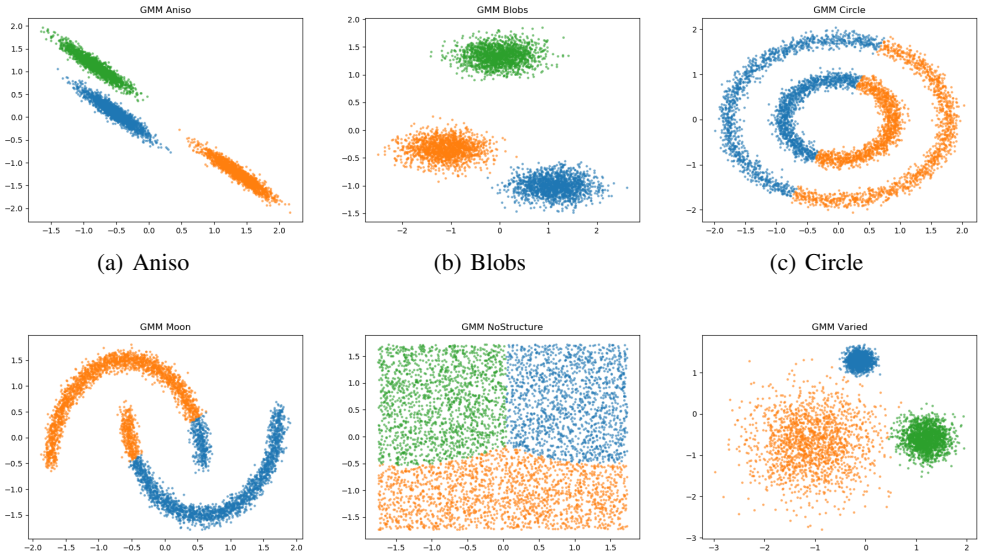


(a) Aniso (b) Blobs (c) Circle

(d) Moon (e) No Structure (f) Varied

Figure 1: Spectral Clustering on Different Datasets

We can also observe 5 that Gaussian Mixture Model works quite well on Aniso, Blobs and Varied datasets. It prefers data that distributes in clusters which is regular and sparse without strange structures. It's because GMM is a mixture of Gaussian distributions, which can easily fit those "ellipses".



(a) Aniso (b) Blobs (c) Circle

(d) Moon (e) No Structure (f) Varied

Figure 2: GMM Clustering on Different Datasets